Autorzy:

- Anna Krzaczkowska gr. 2,
- Szymon Trofimiec gr. 3,
- Urszula Pilśniak gr. 2.

Tematyka projektu

Naszym projektem jest baza danych szkółki narciarsko - snowboardowej z interfejsem podłączonym do tej bazy, dzięki któremu w wygodny sposób będzie można zarządzać harmonogramem, klientami i instruktorami.

Cele

Celem jest stworzenie aplikacji (wykorzystującej bazę danych), która pozwala na sprawne wpisywanie danych do systemu zachowując spójność bazy (kontrola zarówno z poziomu aplikacji, jak i bazy danych). Planowane ograniczenia wpisywania:

- nie można przyporządkować instruktorowi grupy, w dniach w których nie pracuje,
- nie można ustalić instruktorowi zajęć w godzinach w których już jakieś ma lub jest nieobecny,
- nie można do grupy o pewnym poziomie zaawansowania wpisać dziecka które nie ma odpowiedniej odznaki (odznaki opisane są niżej),
- instruktor nie może mieć żadnych zajęć w czasie kiedy nie jest ubezpieczony,
- wpisując krotkę do harmonogramu musi być albo nieobecnosc na true i wtedy id_grupy i id_klienta na NULL, albo dokładnie jedno z nich dwóch inne niż NULL, a nieobecnosc na false - to pozwoli na rozróżnienie zajęć grupowych od indywidualnych i nieobecności,
- nie można zapisać dzieci do grupy która jest pełna, oraz nie można wpisać grupy do harmonogramu jeśli nie ma do niej zapisanej wystarczająco dużo osób.

Wprowadzeniem mechanizmów kontrolujących powyższe ograniczenia zajmiemy się w drugim etapie pracy nad projektem.

W etapie drugim dodaliśmy wszystkie powyższe funkcjonalności.

Funkcjonalności aplikacji:

- dodawanie klientów
- wprowadzanie zajęć i nieobecności do harmonogramu,
- tworzenie grup oraz aktualizacja odznak dzieci
- wyświetlanie w estetyczny sposób harmonogramu dziennego,
- obliczanie wypłaty dziennej każdego pracownika.
- przyznawanie odznak
- dodawanie do grup
- Wyświetlanie poczekalni, aktualnie dostępnych grup

Opis schematu bazy:

Tabele:

- **Ubezpieczenia**: każdy instruktor narciarski/snowboardowy musi mieć ważne ubezpieczenie, aby móc prowadzić zajęcia. W tej tabeli przechowywany jest numer polisy, oraz daty między którymi ubezpieczenie jest ważne. Każdy instruktor ma przypisaną do siebie co najmniej jedna krotkę (każdy instruktor przed zatrudnieniem musi mieć już wykupione ubezpieczenie), ale może mieć też kilka ubezpieczeń na raz.
- **Instruktorzy**: ta tabela zawiera podstawowe dane o instruktorach: imię, nazwisko i numer telefonu.
- **Stopnie**: zawierają informacje o stopniach instruktorskich (oddzielne stopnie dla instruktorów narciarstwa i snowboardu).
- **Stawki_stopnie:** zawiera informacje o aktualnych stawkach stopni instruktorskich i historii ich zmian.
- **Dostępnosc_sezon**: opisuje ona w jakich dniach instruktorzy zadeklarowali swoją obecność.
- **Grupy:** ta tabela opisuje grupy zajęciowe szkółki. W zależności od poziomu zaawansowania zajęć dzieci, które ukończyły grupę otrzymują odpowiednie odznaki pozwalające na kontrolowanie ich rozwoju oraz zapisywanie się do grup o większym stopniu zaawansowania - nie każdy dziecko wpisane na grupę musi dostać odznakę.
- **Harmonogram**: największa tabela tej bazy danych. Opisuje harmonogram wszystkich zajęć odbywających się w szkółce, w tym zajęcia indywidualne i grupowe. Dodatkowo tabela uwzględnia godziny w których instruktorzy mają przerwy/są nieobecni (zakładamy, że w dniach podanych w tabeli Dostępnosc_sezon jeśli nie jest podane inaczej instruktorzy pracują od 9 do 20.
- **Klienci**: tabela opisująca klientów szkółki i zawierająca podstawowe dane o nich (osoby poniżej 18 roku życia mają zawarty numer do rodzica).
- **Odznaki:** Odznaka wyznacza poziom zaawansowania osoby w jeżdżeniu na nartach/snowboardzie. Nową odznakę otrzymuje się po pomyślnym ukończeniu grupy o danym stopniu zaawansowania (nie każdy uczestnik grupy otrzymuje odznakę). Odznaki również można przenieść z innej szkółki narciarskiej, bądź mogą być przyznane na podstawie opinii instruktora ze szkółki (aby umożliwić zapisanie dziecka do grupy innej niż początkująca).
- **Lista_oczekujacych:** do tej tabeli wstawiane są dzieci, które chcą dołączyć do grupy, ale jeszcze taka nie istnieje.
- **Sporty:** tabela zawiera sporty dostępne w naszej szkółce(aktualnie narty i snowboard).
- **Reszta tabel** (instruktorzy_stopnie, dzieci_odznaki, dzieci_grupy) definiują relacje miedzy tabelami.

Wyzwalacze:

- **b_grupy_insert:** Wyzwalacz sprawdza czy jest wystarczająco dzieci w liście oczekujących aby utworzyć grupe, oraz nie pozwala na dodanie grupy, która rozpoczyna się w przeszłości,
- **a_grupy_insert:** przepisuje dzieci z listy_oczekujacych do tabeli dzieci_grupy,
- **lista_oczekujacych_ins:** nie pozwala na dodawanie dzieci do grupy, jeśli jest w innej która zaczyna się w tym samym terminie i ma tą samą odznakę (w szczególności dziecko nie może czekać na grupę, w której już jest), sprawdza, czy dziecko ma odpowiednią odznakę aby zostać wpisanym do grupy, oraz jeśli istnieje jakaś grupa o tych parametrach, to zostaje do niej automatycznie przypisane,
- **harmonogram_add:** dba o spójność harmonogramu, w tym dostępności instruktorów, nachodzenie się zajęć dla instruktorów. Sprawdza też czy instruktor uczy tego sportu i czy jest wykwalifikowany do prowadzenia zajęć grupowych,
- **dzieci_grupy_del:** jeśli w liscie_oczekujacych jest ktoś chętny do tej grupy, to po usunięciu dziecka dodajemy je do tej grupy. Dodatkowo jeśli po usunięciu dziecka z grupy ilość dzieci w grupie wyniesie 0, to grupa zostaje usunięta,
- **dostepnosc_sezon_tr:** przy dodaniu dostępności sprawdza ,czy przez cały okres dostępności instruktor jest ubezpieczony,
- **odznaki_immutability:** zabiera możliwość wpływania na tabele odznaki
- **dzieci_grupy_insert:** sprawdza czy dziecko ma odpowiednia odznake, aby dołączyć do grupy,
- **harmonogram_upd:** nie pozwala na zmienianie i wstawianie krotek do harmonogramu przed currentdate(),
- **harmonogram_del:** nie pozwala na usuwanie krotek z harmonogramu przed currentdate().

Reguły:

Wszystkie reguły blokują dodawanie/updatowanie/usuwanie,
bardziej skomplikowane operacje wykonujemy za pomocą wyzwalaczy.

Indeksy:

- **klienci_idx :** indeks na imię i nazwisko klienta,
- **instruktorzy_idx** : indeks na imię instruktora.

Funkcje:

- **wstaw_klienta** - wstawia klienta do tabeli klienci,
- **wstaw_nieobecności** - wstawia nieobecność godzinową instruktora do harmonogramu,
- **umow_konkretny** - umawia klienta na zajęcia do podanego instruktora i wstawia krotkę do harmonogramu
- id_klienta - wypisuje dane o kliencie (w tym zdobyte odznaki) o podanym imieniu i nazwisku,
- **currentdate()** - zwraca datę '2024-01-09' jest potrzebna do imitacji prawdziwej szkółki, w której nie można usuwać ani zmieniać krotek z harmonogramu przed current_date,
- **wyswietl_harmonogram**(dzien date) - wyświetla harmonogram na dany dzień,
- **wyplata** - oblicza wypłatę z danego dnia dla danego instruktora,
- **umow_dowolny** - umawia klienta na zajęcia do wolnego instruktora, który w danym dniu ma najmniej lekcji i wpisuje zajęcia do harmonogramu,
- **max_stopien** - zwraca najwyższy posiadany przez danego instruktora stopień w danym sporcie,
- **znajdz_instruktora** - zwraca tabelę instruktorów o danym imieniu,
- **ile_dzieci** - zwraca ile jest dzieci w danej grupie,
- **instruktor_grupa** - zwraca jaki instruktor zajmuje się daną grupą,
- **wyswietl_grupy** - wyświetla wszystkie dostępne grupy zaczynające się po wybranej dacie - czyli takie do których można dodać jeszcze dzieci o ile nie są pełne,
- **wyswietl_dzieci_grupy** - wyświetla dane o klientach w danej grupie,
- **wyswietl_poczekalnie** (id_odz int, dzien date) - wyświetla poczekalnię z ludźmi którzy chcą być w grupie na danym poziomie i w tym samym czasie,
- **wyswietl_poczekalnie**() - wyświetla aktualną listę oczekujących na grupę,
- **max_odznaka** (id_klienta int, id_sportu int) - zwraca największą odznakę jaką posiada klient w danym sporcie,
- **dodaj_do_grupy**(id_klienta int, id_odznaki int, data_rozpoczecia date) - dodaje dziecko do poczekalni, reszta czynności obsługiwana przez wyzwalacze,
- **dodaj_grupe**(instruktor int, id_odzn int, data_rozpoczecia date, maks_dzieci int, min_dzieci int) - tworzy grupe o danych parametrach i tworzy zajęcia od 9 do 12 na następne 5 dni,
- **licznosc_grupy**(id grupy int) - zwraca ile dzieci jest w grupie o danym indeksie
- **nadaj_odznake**(id_klienta int, id_odznaki int, data_uzysk date) - nadaje dziecku odznake,
- **archiwizuj_liste_oczekujacych** - usuwa krotki z tabeli sprzed currentdate,
- **deactivate_and_delete** - dezaktywuje na moment trigger nie pozwalający na usuwanie krotek z harmonogramu sprzed currentdate i usuwa wszystkie które były rok i więcej przed currentdate.

Napotkane problemy:

- ➢ Problemy z harmonogramem:
  - Jak przechowywać informacje, w jakich godzinach dostępny jest instruktor? Rozwiązanie : dodanie pola czy_nieobecność pozwalającego na wpisywanie nagłych i/lub krótkich, kilkugodzinnych nieobecności do harmonogramu.
  - Jak jednocześnie przechowywać informacje o zajęciach grupowych i indywidualnych?
    Rozwiązanie: w zależności od rodzaju zajęć id_grupy lub id_klienta jest null. W ten sposób łatwo określić dostępność danego instruktora bez łączenia tabel.
  - Aktualizacja harmonogramu, co jeśli po wpisaniu zajęć do harmonogramu instruktor będzie musiał opuścić te zajęcia? Są to sytuacje wyjątkowe i będą występowały na tyle rzadko, że muszą być rozstrzygane indywidualnie (np. jeśli nie ma dostępnych innych instruktorów trzeba dzwonić do klientów/ rodziców dzieci, że zajęcie się nie mogą odbyć) - dodamy w aplikacji odpowiednie opcje, aby móc skasować z harmonogramu zajęcia.

- ➢ Redundancja w tabelach
  - Czemu jest rozdzielenie na instruktorów i klientów skoro obie tabele przechowują te same dane?
    Odpowiedź: Rozdzielenie takie stosuje się w tego typu systemach ze względu na różnice w restrykcjach w zabezpieczeniu danych pracowników i klientów. Instruktor jest wewnętrznym pracownikiem firmy natomiast dane klientów ze względu na RODO muszą być chronione w inny sposób. Dodatkowo połączenie tych tabel wymagałoby wprowadzenia dodatkowego, “sztucznego” rozróżnienia kursanta od instruktora oraz powodowało niepotrzebne komplikacje w związku z kompozycją relacji (duża liczba tabel związanych z instruktorami a niezwiązana z klientami i tak samo duża liczba tabel związana z klientami a niekoniecznie z instruktorami). Zawsze szukając informacji o instruktorze szukalibyśmy wśród wszystkich klientów i na odwrót. Instruktorzy nie będą nigdy klientami.
  - redundancja w tabelach grupy i harmonogram gdzie niepotrzebnie powtarzane było id_instruktora rozwiązaliśmy to usuwając go z tabeli grupy. Teraz grupy nie mają instruktora prowadzącego.
- ➢ Inne
  - Przechowywanie danych pracowników, którzy nie są instruktorami.
    Nie ma potrzeby dodawania ich do bazy danych, ponieważ jest ich niewielu (dwie panie w okienku do zapisywania kursantów, księgowa i szef) i nie łączą się z resztą bazy. Dodanie informacji o nich może wymagać dodania sztucznych krotek czy nowych tabel, które są zupełnie niepowiązane z resztą bazy danych, dlatego zdecydowaliśmy, że nie jest to kluczowe dla naszej bazy.
  - Jak mieć pewność, że mamy wystarczająco dzieci, aby utworzyć grupę.
    Wpadliśmy na 2 rozwiązania - dodawanie grup jest transakcja, a po dodaniu sprawdzamy, czy liczebność grupy jest pomiędzy max a min dzieci. Jednak w rzeczywistości wymaga to notowania chętnych na kartce, i dodawanie

grupy dopiero wtedy, gdy zbierze się odpowiednio dużo osób. A że chcemy naprawiać rzeczywistość, a nie ją odwzorowywać to wpadliśmy na “liste oczekujacych” która pełni funkcje właśnie tej kartki, i automatyzuje ten proces.

Obsługa aplikacji:

Aby uruchomić aplikację najpierw należy przejść do *~/src/ASU/src/main/java/main* i w pliku *User.java* zmienić login, hasło oraz nazwę bazy danych. Następnie należy zaimportować projekt za pomocą *pom.xml* i uruchomić *Menu.java*. Aplikacja w niektórych miejscach nie jest zabezpieczona przed podawaniem niekompletnych bądź nierealnych danych - nie zawsze wyświetla wiadomość, że tak się nie da zrobić, natomiast sama baza danych jest zabezpieczona przed takimi sytuacjami więc dane błędne lub niekompletne nie zostaną do niej dodane - wszystkie przyciski mają strong exception guarantee (stan bazy się nie zmiania i aplikacja działa dalej).