**Temat: Portal kinowy z funkcją rezerwowania miejsc i zakupu biletów.**

Autorzy: Maciej Umiński, Mateusz Sondej

**Opis**

Nasz projekt to portal kinowy o nazwie „M&M” zintegrowany z bazą danych, umożliwiający użytkownikom zakup biletów na wybrane seanse filmowe przez stronę internetową i dokonywanie rezerwacji miejsc w wybranej sali kinowej. Funkcjonalność systemu obejmuje możliwość korzystania z voucherów, wyboru miejsc VIP w sali kinowej, zwiększając tym samym dostosowanie oferty do indywidualnych preferencji klientów. Opcją dla klienta będzie również wybór przekąsek na daną rezerwację. System umożliwia na przykład przypisanie promocji do konkretnych biletów czy zarządzanie harmonogramem seansów, dzięki czemu kino może efektywnie odpowiadać na potrzeby widzów i zoptymalizować wykorzystanie sal projekcyjnych. Relacje jeden-do-wielu oraz wiele-do-wielu pomiędzy tabelami takimi jak FILMS, SEANCES, SCREENING\_ROOMS, RESERVATIONS czy TICKETS zapewniają pełną skalowalność i elastyczność systemu. Dzięki przemyślanym ograniczeniom, takim jak unikalność danych, czasowe ograniczenia rezerwacji czy ograniczona liczba miejsc VIP, portal zapewnia wysoką jakość usług i zachowuje standardy doświadczeń kinowych.

**Technologie zaimplementowane do projektu**

**1. PHP Framework - Laravel**

- Laravel 11.x jako główny framework do budowy aplikacji webowej.

**2. Bazodanowe**

- Oracle Database 19c jako główna baza danych do przechowywania informacji o filmach, seansach, użytkownikach i rezerwacjach.

- PL/SQL używane do tworzenia procedur składowanych, triggerów oraz zarządzania bazą danych.

**3. Frontend**

- HTML5 do struktury strony.

- CSS3 do stylizacji strony.

- Bootstrap 5 jako framework CSS do tworzenia responsywnych i nowoczesnych interfejsów użytkownika.

**4. Backend**

- PHP do logiki serwera.

- Laravel Blade jako silnik szablonów do renderowania dynamicznych stron HTML.

**5. Narzędzia developerskie**

- Git jako system kontroli wersji do zarządzania kodem źródłowym.

- Composer do zarządzania zależnościami PHP.

**6. JavaScript**

- Używane do dynamicznych interakcji i walidacji po stronie klienta.

**Założenia**

1. **Elastyczność seansów i filmów**: System zakłada możliwość łatwego dodawania, modyfikowania i usuwania filmów oraz planowania seansów, aby umożliwić kinu dynamiczne reagowanie na zmiany w repertuarze oraz preferencje widzów.
2. **Możliwość rezerwacji i zakupu online**: Użytkownicy będą mogli dokonywać rezerwacji miejsc oraz zakupu biletów online przez portal kinowy.
3. **Personalizacja oferty**: System umożliwia wykorzystanie vouchera na dany bilet oraz możliwość wyboru miejsc VIP.
4. **Automatyzacja komunikacji**: Komunikacja z klientem, w tym rezerwacja miejsc i zakup biletów, jest realizowana poprzez stronę internetową kina.
5. **Kaskadowe usuwanie się rekordów**: Jeżeli jakiś rekord zostanie usunięty z tabeli z bazy danych, to wszystkie rekordy powiązane z tym rekordem także zostaną usunięte.
6. **Brak możliwości usunięcia administratora**: Nie można usunąć administratora, także sam admin siebie nie może usunąć na stronie.
7. **Automatycznie tworzone miejsca dla nowych sal**: Kiedy admin stworzy nową salę kinową, to automatycznie są tworzone nowe miejsca dla tej sali.

**Ograniczenia**

1. **Jedno miejsce na jeden seans**: System musi zapewniać, że jedno miejsce nie zostanie zarezerwowane lub sprzedane więcej niż jednemu klientowi na ten sam seans.
2. **Unikalność danych**: Każde ID (np. Film\_ID, Seance\_ID, User\_ID) musi być unikalne, aby zapewnić jednoznaczną identyfikację rekordów w bazie danych.
3. **Czasowe ograniczenie rezerwacji**: Rezerwacje miejsc mogą być dokonywane do 30 minut przed rozpoczęciem seansu.
4. **Ograniczona liczba miejsc VIP**: Miejsca VIP są oferowane w ograniczonej liczbie dla poszczególnych sal.
5. **Promocje czasowe**: Promocje i zniżki mogą być stosowane tylko w określonych ramach czasowych i zgodnie z zasadami określonymi przez kino, co wymaga ścisłego monitorowania terminów ważności promocji.

**Diagram ERD**

**Obraz zawierający tekst, zrzut ekranu, diagram, Plan

Opis wygenerowany automatycznie**

**Struktura bazy danych**

* **Films:** ID (PK), NAME, DESCRIPTION, DURATION, GENRE, IMG
* **Seances:** ID (PK), FILM\_ID (FK), SCREENING\_ROOM\_ID (FK), WORKER\_ID (FK), TECHNOLOGY\_ID (FK), PROMOTION\_ID (FK), START\_TIME, END\_TIME
* **Screening\_Room:** ID (PK), SEATS, NUMBER\_OF\_SEATS, PRICE\_FOR\_SEAT
* **Technologies:** ID (PK), NAME
* **Reservations:** ID (PK), SEANCE\_ID (FK), USER\_ID (FK)
* **Seats:** ID (PK), SCREENING\_ROOM\_ID (FK), ROW\_NUMBER, SEAT\_IN\_ROW, VIP
* **Reservation\_Seats:** ID (PK), RESERVATION\_ID (FK), SEAT\_ID (FK)
* **Tickets:** ID (PK), RESERVATION\_ID (FK), VOUCHER\_ID (FK), PRICE
* **Users:** ID (PK), NAME, SURNAME, EMAIL, PHONE\_NUMBER, ROLE, PASSWORD
* **Promotions:** ID (PK), DISCOUNT, START\_TIME, END\_TIME
* **Workers:** ID (PK), NAME, SURNAME
* **Products:** ID (PK), NAME, PRICE
* **Reservation\_Products:** ID (PK), PRODUCT\_ID (FK), RESERVATION\_ID (FK)
* **Vouchers:** ID (PK), NAME, DISCOUNT

**Opis relacji:**

Użytkownicy mogą skorzystać z różnorodnych promocji, które są przypisane do konkretnych seansów, co jest możliwe dzięki relacji jeden-do-wielu [1:N] między tabelami **Promotions i Seances**, gdzie każdy seans może być związany z określoną promocją w danym dniu. Ponadto, każdy seans jest obsługiwany przez wyznaczonych pracowników (relacja jeden-do-wielu [1:N] między **Workers a Seances**), co gwarantuje utrzymanie porządku i wysoką jakość usług. Wdrażane są również rozmaite technologie wyświetlania obrazu, takie jak 3D czy IMAX, dzięki relacji jeden-do-wielu [1:N] między **Technologies a Seances**, co pozwala na dopasowanie technologii projekcji do danego filmu. Dla zwiększenia komfortu użytkowników, portal oferuje także możliwość zakupu przekąsek i napojów do konsumpcji podczas seansu, co jest wynikiem zastosowania relacji wiele-do-wielu [N:N] między tabelami **Products i Reservations**, umożliwiającej przypisanie wielu produktów spożywczych do jednej rezerwacji. Każdy film może być przypisany do wielu seansów dzięki relacji jeden-do-wielu [1:N] między **Films a Seances**, co pozwala na wielokrotne wyświetlenie tego samego filmu w różnych terminach i godzinach. Relacja jeden-do-wielu [1:N] między **Screening\_Rooms a Seances** umożliwia zarządzanie harmonogramem pokazów dla poszczególnych sal. Przekłada się to na wydajne wykorzystanie możliwości sal kinowych. Dzięki relacji jeden-do-wielu [1:N] między **Seances i Reservations** możliwe jest śledzenie, które rezerwacje dotyczą danego seansu, co jest kluczowe dla zarządzania obłożeniem sali. Każdy klient może mieć wiele rezerwacji dzięki relacji jeden-do-wielu [1:N] między tabelami **Users i Reservations**. Każda rezerwacja w tabeli **Reservations** posiada klucz obcy USER\_ID, co umożliwia śledzenie wszystkich rezerwacji przypisanych do jednego klienta. Jedna rezerwacja (np. dla grupy ludzi na jeden seans) może obejmować wiele biletów, dzięki relacji jeden-do-wielu [1:N] między **Reservations i Tickets**. Relacja wiele-do-wielu [N:N] między tabelami **Seats i** **Reservations**, w której wiele miejsc może być zarezerwowanych przez różne rezerwacje, a każda rezerwacja może obejmować różne miejsca, dzięki czemu dany klient może zarezerwować wiele miejsc na dany seans. Relacja jeden-do-wielu [1:N] między **Screening\_Rooms a Seats** umożliwia określenie ile dana sala ma miejsc, dzięki kluczowi obcemu SCREENING\_ROOM\_ID w tabeli Seats. Dla każdego biletu może być przypisany odpowiedni voucher, dzięki relacji jeden-do-wielu [1:N] między tabelami **Vouchers i Tickets** – reprezentowanej przez unikalny klucz VOUCHER\_ID w tabeli **Tickets**. Taki kompleksowy system relacji między tabelami zapewnia efektywne zarządzanie danymi i oferuje użytkownikom kinowym pełny zakres usług, podnosząc standardy doświadczeń kinowych.

**Relacje między tabelami:**

* + 1. FILMS <-> SEANCES:

o Relacja jeden-do-wielu (1:N). Każdy film może być pokazywany w ramach wielu seansów. Klucz obcy w tabeli SEANCES to FILM\_ID.

* 1. SCREENING\_ROOMS <-> SEANCES:

o Relacja jeden-do-wielu (1:N). Każda sala kinowa może być miejscem dla wielu seansów. Klucz obcy w tabeli SEANCES to SCREENING\_ROOM\_ID.

* 1. TECHNOLOGIES <-> SEANCES:

o Relacja jeden-do-wielu (1:N). Każda technologia (np. 3D, IMAX) może być wykorzystywana w wielu seansach. Klucz obcy w tabeli SEANCES to TECHNOLOGY\_ID.

* 1. SEANCES <-> RESERVATIONS:

o Relacja jeden-do-wielu (1:N). Każdy seans może mieć wiele rezerwacji. Klucz obcy w tabeli RESERVATIONS to SEANCE\_ID.

* 1. USERS <-> RESERVATIONS:

o Relacja jeden-do-wielu (1:N). Jeden użytkownik może dokonać wielu rezerwacji. Klucz obcy w tabeli RESERVATIONS to USER\_ID.

* 1. RESERVATIONS <-> TICKETS:

o Relacja jeden-do-wielu (1:N). Jedna rezerwacja może obejmować wiele biletów. Klucz obcy w tabeli TICKETS to RESERVATION\_ID.

* 1. SEATS <-> RESERVATIONS:

o Relacja wiele-do-wielu (N:N). Każda rezerwacja może mieć przypisane wiele miejsc, a każde miejsce może być przypisane w wielu rezerwacjach.

* 1. SCREENING\_ROOMS <-> SEATS:

o Relacja jeden-do-wielu (1:N). Każda sala kinowa zawiera wiele miejsc. klucz obcy w tabeli SEATS to SCREENING\_ROOM\_ID.

* 1. VOUCHERS <-> TICKETS:

o Relacja jeden-do-wielu (1:N). Każdy bilet może korzystać z jednego vouchera. Klucz obcy w tabeli TICKETS to VOUCHER\_ID.

* 1. PROMOTIONS <-> SEANECS:

o Relacja jeden-do-wielu (1:N). Na każdy seans może być wykorzystana promocja. Klucz obcy w tabeli SEANCES to PROMOTION\_ID.

* 1. WORKERS <-> SEANCES:

o Relacja jeden-do-wielu (1:N). Do każdego seansu można przypisać jednego pracownika. Klucz obcy w tabeli SEANCES to WORKER\_ID.

* 1. PRODUCTS <-> RESERVATIONS:

o Relacja wiele-do-wielu (N:N). Dla każdej rezerwacji mogą być przypisane wiele produktów spożywczych (przekąsek), a każdy przekąsek może być przypisany do wielu rezerwacji.

**Funkcjonalności:**

1. **Przeglądanie repertuaru**: Możliwość wyświetlania aktualnego repertuaru filmowego, włączając w to szczegółowe informacje o filmach (nazwa, opis, czas trwania, gatunek).
2. **Rezerwacja i zakup biletów**: Funkcja umożliwiająca rezerwację miejsc na wybrany seans oraz zakup biletów, w tym wybór miejsca z dostępnej mapy miejsc w sali kinowej.
3. **Możliwość zalogowania się**: Użytkownicy mogą zalogować się na stronie kina.
4. **Wykorzystanie vouchera**: Możliwość użycia zniżki przy kupnie biletu za pomocą vouchera.
5. **Zakup przekąsek**: Opcja dodania do zamówienia przekąsek i napojów, które mogą być odbierane w kinie przed seansem.
6. **Zarządzanie rezerwacjami**: Funkcja umożliwiająca użytkownikom przeglądanie swoich aktualnych rezerwacji, a także anulowanie rezerwacji.
7. **Możliwość edycji własnych danych**: Użytkownicy mogą zmieniać swoje dane osobowe.
8. **Możliwość manipulowania danymi w bazie danych przez admina**: Admin ma dostęp do podstron z tabelami gdzie może wykonywać operacje CRUD na tychże tabelach.

**Baza danych**

**Przykładowy CRUD:**

**Obraz zawierający tekst, Czcionka, numer, oprogramowanie

Opis wygenerowany automatycznieDla tabeli FILMS:**

CRUD dla tabeli FILMS jak i innych tabel został wykonany za pomocą procedur PL/SQL.

Dodawanie filmów.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Ta procedura pozwala na odczytanie danych o filmie z tabeli FILMS na podstawie jego ID.

Opis:

* p\_name: Nazwa filmu.
* p\_description: Opis filmu.
* p\_duration: Czas trwania filmu.
* p\_genre: Gatunek filmu.
* p\_img: Ścieżka do obrazu filmu.

Usuwanie filmów.

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

Ta procedura pozwala na usunięcie filmu z tabeli FILMS na podstawie jego ID.

Opis:

* p\_film\_id: ID filmu, który ma zostać usunięty.

Wczytywanie filmów.

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, Czcionka

Opis wygenerowany automatycznie

Ta procedura pozwala na odczytanie danych o filmie z tabeli FILMS na podstawie jego ID.

Zawarty jest w niej również kursor ‘SYS\_REFCURSOR’.

Opis:

* p\_film\_id: ID filmu.
* p\_film: Zmienna kursorowa, która zwraca dane filmu.

Edycja filmów.

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Ta procedura pozwala na aktualizację danych filmu w tabeli FILMS.

Opis:

* p\_film\_id: ID filmu.
* p\_name: Nowa nazwa filmu.
* p\_description: Nowy opis filmu.
* p\_duration: Nowy czas trwania filmu.
* p\_genre: Nowy gatunek filmu.
* p\_img: Nowa ścieżka do obrazu filmu.

Procedury PL/SQL powyżej realizują pełną funkcjonalność CRUD dla tabeli FILMS, pozwalając na tworzenie, odczytywanie, aktualizowanie i usuwanie rekordów w tej tabeli. Każda z procedur jest zaprojektowana z uwzględnieniem odpowiednich parametrów wejściowych i wyjściowych, co umożliwia zarządzanie danymi filmowymi w bazie danych.

**Triggery:**

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

**Trigger ‘trg\_check\_seance\_overlap’** zapobiega nakładaniu się seansów w tej samej sali kinowej. Działa przed każdą operacją wstawiania lub aktualizacji rekordu w tabeli ‘seances’, sprawdzając, czy terminy nowego lub modyfikowanego seansu nie kolidują z istniejącymi seansami w tej samej sali. Jeśli wykryje kolizję, generuje błąd aplikacji, uniemożliwiając zapis konfliktującego seansu. Dzięki temu zapewnione jest, że seanse w tej samej sali kinowej nie będą się czasowo nakładać.

Obraz zawierający tekst, zrzut ekranu, dokument, numer

Opis wygenerowany automatycznie

**Trigger ‘trg\_check\_duplicate\_reservation\_product’** zapobiega dodawaniu lub aktualizacji rekordu w tabeli ‘reservation\_products’ z tym samym produktem dla tej samej rezerwacji więcej niż jeden raz. Działa przed operacjami wstawiania i aktualizacji. Sprawdza, czy kombinacja ‘product\_id’ i ‘reservation\_id’ już istnieje. Jeśli tak, generuje błąd aplikacji, informując, że produkt jest już zarezerwowany dla tej rezerwacji, co zapobiega zduplikowanym wpisom.

Obraz zawierający tekst, zrzut ekranu, numer

Opis wygenerowany automatycznie

**Trigger ‘trg\_check\_duplicate\_reservation\_seat’** zapobiega dodawaniu lub aktualizacji rekordu w tabeli ‘reservation\_seats’ z tym samym miejscem dla tej samej rezerwacji więcej niż jeden raz. Działa przed operacjami wstawiania i aktualizacji. Sprawdza, czy kombinacja ‘seat\_id’ i ‘reservation\_id’ już istnieje. Jeśli tak, generuje błąd aplikacji, informując, że miejsce jest już zarezerwowane dla tej rezerwacji, co zapobiega zduplikowanym wpisom.

Obraz zawierający tekst, zrzut ekranu

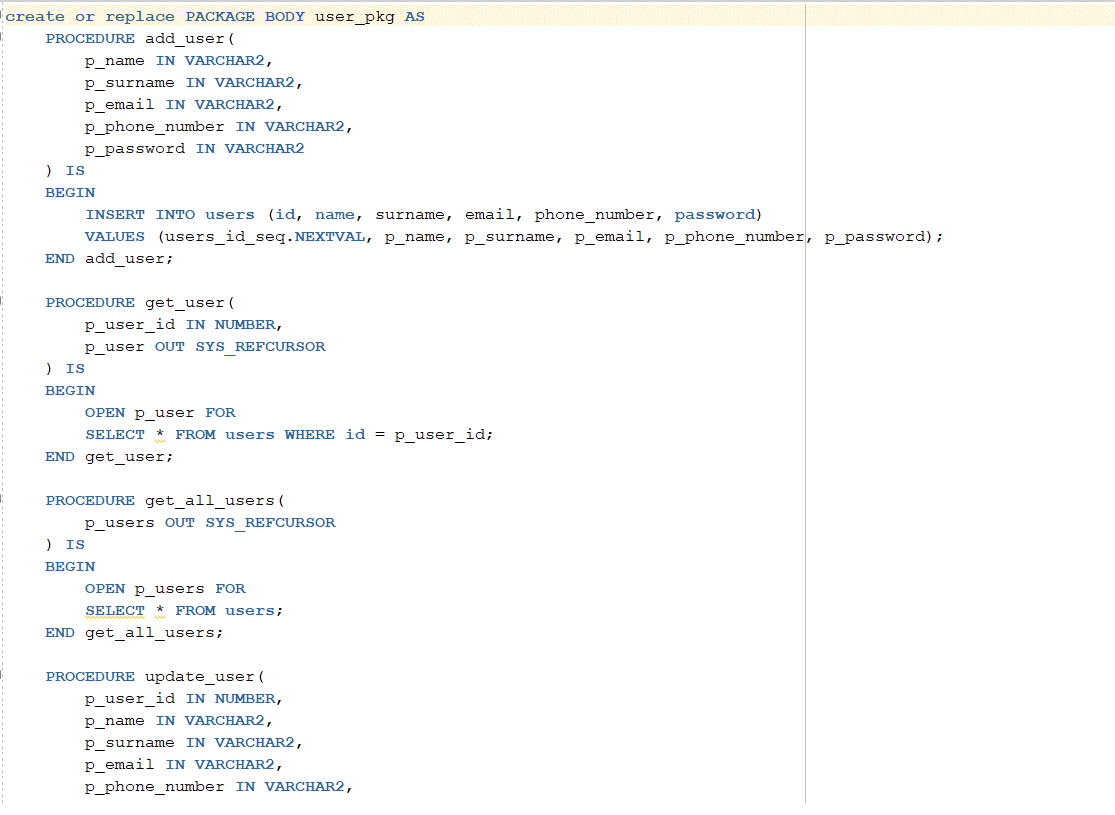
Opis wygenerowany automatycznie

**Trigger ‘trg\_check\_duplicate\_seat’** zapobiega dodawaniu lub aktualizacji rekordu w tabeli ‘seats’ z tym samym numerem rzędu i miejscem w tej samej sali kinowej więcej niż jeden raz. Działa przed operacjami wstawiania i aktualizacji. Sprawdza, czy kombinacja ‘screening\_room\_id, row\_number’ i ‘seat\_in\_row’ już istnieje. Jeśli tak, generuje błąd aplikacji, informując, że miejsce o tym samym numerze rzędu i miejscu w rzędzie już istnieje w tej sali kinowej, co zapobiega zduplikowanym wpisom.

**Pakiety:**

Obraz zawierający tekst, zrzut ekranu, dokument, Czcionka

Opis wygenerowany automatycznie



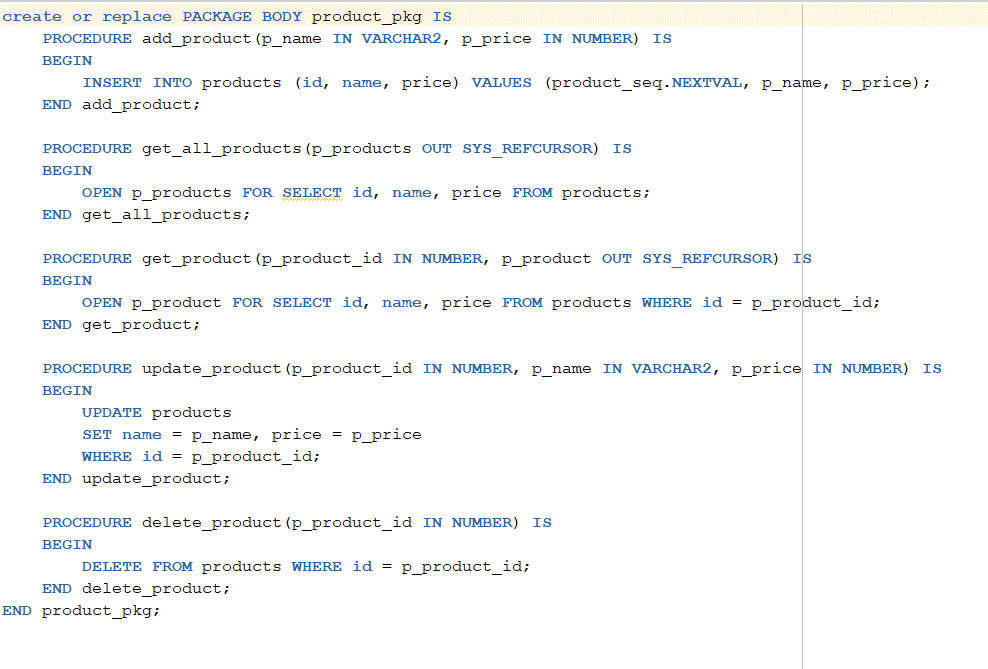
Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

**Pakiet ‘user\_pkg’** zawiera procedury umożliwiające zarządzanie użytkownikami w bazie danych. Procedura ‘add\_user’ umożliwia dodanie nowego użytkownika do tabeli ‘users’, wstawiając jego dane takie jak imię, nazwisko, email, numer telefonu i hasło. Procedura ‘get\_user’ pozwala na pobranie informacji o konkretnym użytkowniku na podstawie jego ID, wykorzystując kursor do zwrócenia wyników. Procedura ‘get\_all\_users’ zwraca informacje o wszystkich użytkownikach z tabeli ‘users’, również wykorzystując kursor. Procedura ‘update\_user’ umożliwia aktualizację danych istniejącego użytkownika na podstawie jego ID, zmieniając imię, nazwisko, email, numer telefonu i hasło. Procedura ‘delete\_user’ usuwa użytkownika z tabeli ‘users’ na podstawie jego ID.

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie



**Pakiet ‘product\_pkg’** zawiera procedury umożliwiające zarządzanie produktami w bazie danych. Procedura ‘add\_product’ umożliwia dodanie nowego produktu do tabeli products, wstawiając jego nazwę oraz cenę. Procedura ‘get\_all\_products’ zwraca informacje o wszystkich produktach z tabeli products, wykorzystując kursor do zwrócenia wyników. Procedura ‘get\_product’ pozwala na pobranie informacji o konkretnym produkcie na podstawie jego ID, również wykorzystując kursor. Procedura ‘update\_product’ umożliwia aktualizację danych istniejącego produktu na podstawie jego ID, zmieniając nazwę oraz cenę produktu. Procedura ‘delete\_product’ usuwa produkt z tabeli products na podstawie jego ID.

**Funkcje:**

Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

**Funkcja ‘calculate\_ticket\_price’** oblicza cenę biletu na podstawie ID seansu oraz ID miejsca. Zwraca dane jako tablicę rekordów zawierających informacje o podstawowej cenie, rabacie oraz ostatecznej cenie biletu. Najpierw, funkcja określa podstawową cenę biletu na podstawie ceny miejsca w sali kinowej oraz ewentualnego statusu VIP miejsca (cena jest podwajana dla miejsc VIP). Następnie oblicza wartość rabatu, jeśli promocja jest dostępna, uwzględniając rabat jako procent ceny biletu. Ostateczna cena biletu jest obliczana przez odjęcie wartości rabatu od podstawowej ceny. Dane te są pobierane z tabel ‘seats’, ‘screening\_rooms’, ‘seances’ oraz ‘promotions’, korzystając z odpowiednich kluczy obcych w celu powiązania danych. Na koniec, funkcja zwraca wynik w postaci rekordu zawierającego podstawową cenę, wartość rabatu oraz ostateczną cenę biletu.

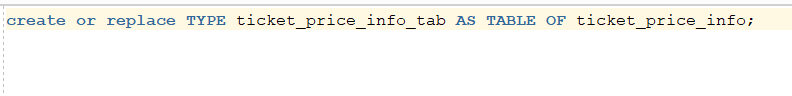
**Typy obiektowe:**

Obraz zawierający tekst, zrzut ekranu, Czcionka, linia

Opis wygenerowany automatycznie

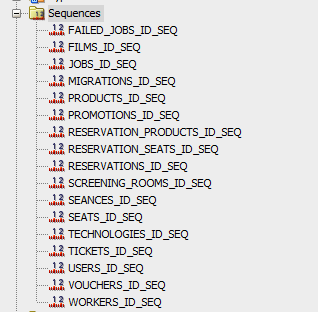
**Obiekt ‘ticket\_price\_info’** jest typem zdefiniowanym w bazie danych, który zawiera trzy atrybuty: ‘base\_price’, ‘discount’ oraz ‘final\_price’. Atrybut ‘base\_price’ przechowuje podstawową cenę biletu przed uwzględnieniem jakichkolwiek rabatów. Atrybut ‘discount’ przechowuje wartość rabatu, który może być zastosowany do ceny biletu w zależności od dostępnych promocji. Atrybut ‘final\_price’ przechowuje ostateczną cenę biletu po uwzględnieniu rabatu. Ten obiekt jest używany w funkcji ‘calculate\_ticket\_price’ do zwracania złożonych wyników obliczeń cen biletów, co umożliwia bardziej strukturalne i zorganizowane przechowywanie tych informacji w ramach jednego typu danych. Dzięki temu kod bazy danych jest bardziej czytelny i zrozumiały, ponieważ wszelkie informacje o cenie biletu są zgrupowane w jednym obiekcie, a nie rozproszone po różnych zmiennych lub tabelach.

**Typy tablicowe:**



**Typ ‘ticket\_price\_info\_tab’** to zdefiniowany typ tabelaryczny, który jest tablicą obiektów typu ‘ticket\_price\_info’. Obiekt ‘ticket\_price\_info’ zawiera trzy atrybuty: ‘base\_price’, ‘discount’ oraz ‘final\_price’, które przechowują odpowiednio podstawową cenę biletu, wartość rabatu oraz ostateczną cenę biletu po uwzględnieniu rabatu. Definicja tego typu tabelarycznego pozwala na przechowywanie i manipulowanie kolekcją obiektów ‘ticket\_price\_info’ jako jednej struktury danych. Typ ‘ticket\_price\_info\_tab’ jest używany w funkcji ‘calculate\_ticket\_price’, która zwraca wyniki obliczeń cen biletów w postaci strumieniowej (pipelined). Dzięki zastosowaniu tego typu, funkcja może zwracać wiele rekordów naraz, co jest bardziej efektywne i czytelne. Umożliwia to również bardziej zorganizowane i strukturalne przetwarzanie danych cen biletów w ramach jednej operacji, ułatwiając zarządzanie wynikami oraz integrację z innymi komponentami aplikacji lub zapytaniami SQL.

**Sekwencje:**



Na zdjęciu widzimy listę sekwencji w bazie danych Oracle, które są używane do generowania unikalnych wartości identyfikatorów dla różnych tabel w bazie danych. Każda sekwencja jest związana z określoną tabelą i służy do zapewnienia, że wartości ID są unikalne i automatycznie zwiększane przy każdym wstawieniu nowego rekordu.

**Procedura:**

Obraz zawierający tekst, zrzut ekranu, wyświetlacz, numer

Opis wygenerowany automatycznie

Powyższy kod **przedstawia procedurę i zadanie** w Oracle, które automatycznie **usuwa stare seanse**. Procedura ‘delete\_old\_seances’ usuwa rekordy z tabeli ‘SEANCES’, gdzie ‘END\_TIME’ jest mniejsze lub równe bieżącej dacie (SYSDATE). Następnie w bloku BEGIN za pomocą ‘DBMS\_SCHEDULER.create\_job’ tworzone jest zadanie o nazwie ‘DELETE\_OLD\_SEANCES\_JOB’, które wykonuje tę procedurę. Zadanie jest typu ‘PLSQL\_BLOCK’, co oznacza, że wykonuje blok PL/SQL. Akcja zadania polega na wywołaniu procedury ‘delete\_old\_seances’. Zadanie rozpoczyna się natychmiast (‘SYSTIMESTAMP’) i powtarza się co minutę (‘FREQ=MINUTELY; INTERVAL=1’). Zadanie jest automatycznie włączone (‘enabled => TRUE’), co zapewnia jego regularne uruchamianie i automatyczne usuwanie przeterminowanych seansów z tabeli.

**Interfejs użytkownika:**

**Dane logowania dla przykładowych użytkowników:**

1. Konto admina:

e-mail: [admin@example.com](mailto:admin@example.com)

hasło: 123

2. Konto użytkownika ‘Jerzy’:

e-mail: [jerzy@example.com](mailto:jerzy@example.com)

hasło: 123

**Widok strony internetowej:**

Na poniższym zdjęciu widać stronę główną naszej strony internetowej kina. Jest tam slider, który można przesuwać w prawo i lewo i zobaczyć wszystkie dostępne filmy.

Obraz zawierający tekst, zrzut ekranu, projekt graficzny, Oprogramowanie multimedialne

Opis wygenerowany automatycznie

Poniżej przedstawiono podstronę ‘repertuar’, gdzie można zobaczyć dostępne seanse.

Obraz zawierający tekst, zrzut ekranu, Reklama internetowa, oprogramowanie

Opis wygenerowany automatycznie

Poniżej zaprezentowano panel logowania.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Po zalogowaniu może wejść na podstronę własnego konta przez zakładkę ‘Konto’.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Po kliknięciu ‘Edytuj dane’ przenosi użytkownika do formularza edycji danych osobowych.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, wyświetlacz

Opis wygenerowany automatycznie

Na podstronie repertuaru, po kliknięciu ‘Kup bilet’ dla danego seansu przenosi nas na podstronę, gdzie można wybrać miejsca w sali kinowej.

Obraz zawierający zrzut ekranu, tekst, wyświetlacz, design

Opis wygenerowany automatycznie

Po wybraniu miejsc i kliknięciu ‘Zarezerwuj’ przenosi nas na podstronę podsumowania rezerwacji. Po kliknięciu ‘Potwierdź Rezerwację’ przenosi nas z powrotem na podstronę repertuarów.

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

Poniżej można zobaczyć, że miejsce jest zarezerwowane na dany seans i nie można je ponownie wybrać, dopóki seans się nie skończy.

Obraz zawierający zrzut ekranu, tekst, design

Opis wygenerowany automatycznie

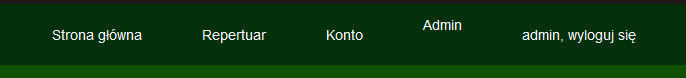
Na podstronie konta użytkownika wyświetla bilety po dokonaniu rezerwacji.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie

Konto admina:

Po zalogowaniu się na konto admina, wyświetla nam dodatkową opcję ‘Admin’ w menu. Admin ma tam możliwość wybrania podstron CRUD dla tabel z bazy danych.



Po wybraniu przykładowej podstrony dla produktów, wyświetla się poniższa podstrona, w formie tabeli dostępne są wszystkie produkty. Można dodawać, usuwać i edytować produkty.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Można dodawać produkty, poniżej pokazany jest formularz.

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

A także można edytować produkty, jak to zaprezentowano poniżej.

Obraz zawierający tekst, zrzut ekranu, Czcionka, oprogramowanie

Opis wygenerowany automatycznie

**Uruchomienie projektu:**

1. Projekt można pobrać ze strony naszego [repozytorium](https://github.com/Matias014/Projekt-kino).

2. Aby uruchomić projekt kina w Laravelu 11.x, używamy pliku start.bat oraz serwera Apache w XAMPP. Plik start.bat zawiera szereg komend, które automatyzują konfigurację i uruchomienie aplikacji.

3. Po uruchomieniu projektu w Visual Studio Code po kliknięciu pliku start.bat i serwera apache w programie XAMPP, trzeba wejść w terminal CMD w Visual Studio Code i uruchomić serwer wpisując komendę ‘*php artisan serve’*.

4. Po tych krokach trzeba wejść w przeglądarkę internetową (np. Google Chrome) i wpisać w pasku wyszukiwania adres ‘*localhost:8000’*. Strona internetowa kina powinna się wyświetlić po tych krokach.

**Literatura i źródła:**

**Strony internetowe i dokumentacje**

1. [**Laravel Documentation**](https://laravel.com/docs/11.x)
   * Oficjalna dokumentacja frameworka Laravel dostarcza pełne i szczegółowe informacje na temat instalacji, konfiguracji oraz używania wszystkich funkcji Laravela. Była to nasza główna referencja podczas tworzenia projektu.
2. [**Oracle Database Documentation**](https://docs.oracle.com/en/database/oracle/oracle-database/19/index.html)
   * Oficjalna dokumentacja Oracle Database 19c była nieocenionym źródłem informacji o konfiguracji, administracji i programowaniu w Oracle Database.
3. [**Bootstrap Documentation**](https://getbootstrap.com/docs/5.3/getting-started/introduction/)
   * Oficjalna dokumentacja frameworka CSS Bootstrap, który został użyty do stylizacji naszego projektu. Zawiera ona kompletny przewodnik po komponentach, układach i najlepszych praktykach.
4. [**Composer**](https://getcomposer.org/doc/)
   * Dokumentacja narzędzia Composer była niezbędna do zarządzania zależnościami PHP w naszym projekcie.
5. [**Git Documentation**](https://git-scm.com/doc)
   * Oficjalna dokumentacja systemu kontroli wersji Git, który używaliśmy do zarządzania kodem źródłowym naszego projektu.
6. [**Kino Radość**](https://kinoradosc.pl/)
   * Oficjalna strona strony internetowej kina „Radość”, który używaliśmy jako inspirację do projektu.
7. [**Kino Helios**](https://helios.pl/rzeszow/kino-helios-galeria/)
   * Oficjalna strona strony internetowej kina „Helios”, który używaliśmy jako inspirację do projektu.