POLITECHNIKA WROCŁAWSKA

WYDZIAŁ ELEKTRONIKI

PROJEKT Z BAZ DANYCH

**BAZA DANYCH ORAZ APLIKACJA WEBOWA DO ZARZĄDZANIA KINEM**

Termin zajęć: Piątek, 9:15–11:00

Autorzy: Prowadzący zajęcia:

Ewa Zajdel, 226494 dr inż. Roman Ptak, W4/K9

Monika Wójcik, 226446

Monika Skoczylas, 195348

Wrocław, 2018 r.

1. Wstęp

1.1. Cel projektu

Celem projektu jest stworzenie bazy danych oraz aplikacji webowej służących do sprzedaży i rezerwacji biletów kinowych oraz do zarządzania kinem.

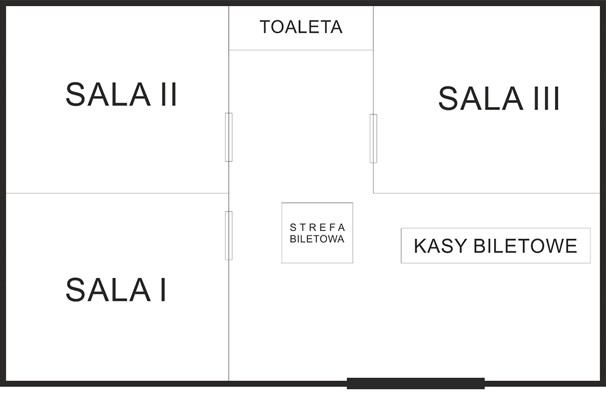
1.2. Zakres projektu

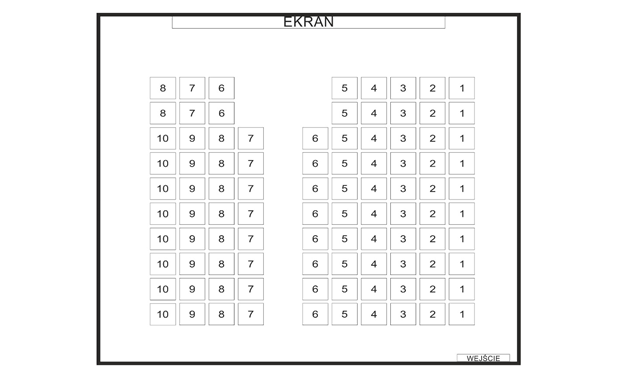
* Analiza wytycznych podanych przez klienta
* Pozyskanie, analiza i implementacja danych
* Stworzenie odpowiedniej bazy danych
* Wdrożenie systemu - dostosowanie oprogramowania do wymagań klienta, testowanie, uruchomienie systemu

2. Analiza wymagań

2.1. Opis działania i schemat logiczny systemu

Struktura składa się z sieci kin znajdujących się w dwóch lokalizacjach. W każdym z nich znajdują się 3 sale, posiadające po około 100 miejsc. Na jedno kino przypada około 20 pracowników, którzy odpowiedzialni są za sprzedaż biletów, nadzór oraz czystość kina. Sale kinowe znajdują się na jednym piętrze. Klient wchodząc do kina przechodzi kolejno przez: kasy biletowe, strefę biletową oraz udaje się do sali. W kasach biletowych znajdują się 3 stanowiska kasowe, których system sprzedaży biletów jest połączony ze sobą tak, aby w tym samym czasie nie można było sprzedać tego samego miejsca na sali. W strefie biletowej znajduje się 2 pracowników, którzy kontrolują klientom bilety oraz nakierowują do określonej sali z seansem. Klient ma możliwość wyboru kina, seansu, rodzaju biletu, a także określonego wolnego miejsca w danej sali. Istnieją różne rodzaje biletów – podzielone ze względu na wiek (ulgowy/studencki/normalny) lub z rabatem w zależności od dnia tygodnia (Tania Środy, Piątkowy Duet). Pracownik nadzoruje sprzedaż biletów, kontroluje rezerwacje miejsc na sali oraz sprawdza bilety przed salą kinową. Kierownik ma dostęp do systemu - kontroluje czas pracy pracownika, może zarządzać użytkownikami systemu oraz salami i repertuarem.

**Rys. 1 Rozkład kina w jednej z lokalizacji.**

**Rys. 2 Przykładowy rozkład miejsc w sali kinowej.**

2.2. Wymagania funkcjonalne

Film opisany jest przez cały szereg własności, m.in. reżyser, gatunek, grupa wiekowa, dla jakiej jest przeznaczony oraz krótki opis filmu. Podstawową funkcjonalnością jest rezerwacja miejsc na dane seanse. Rezerwacja ta odbywa się pod warunkiem istnienia wolnych miejsc w sali, w której odbędzie się seans. Po sfinalizowaniu operacji zakupu bądź rezerwacji zmienia się dostępność wybranego przez użytkownika miejsca. Podczas każdego seansu będzie obecny jeden z pracowników (przypisany do seansu).

Do wszystkich zamówień zrealizowanych przez klientów ma dostęp pracownik kina po zalogowaniu się do systemu indywidualnym loginem i hasłem. Pracownik może również tworzyć zamówienia klientów. Kierownik ma możliwość śledzić czas pracy pracowników - od momentu zalogowania do systemu do momentu wylogowania. Ma dostęp do repertuaru kina i ma możliwość jego modyfikacji. Aby mieć dostęp do danych funkcji, również musi być zalogowany do systemu.

Wymagania są zależne od użytkownika aplikacji. Klient/pracownik/kierownik mają różny dostęp do możliwych funkcji.

2.2.1. Dla klienta

* Wyszukiwanie repertuaru
* Wybór miejsca
* Rezerwacja biletu na wybrany seans bez potrzeby logowania się do systemu

2.2.2. Dla pracownika

* Możliwości klienta
* Logowanie się do systemu za pomocą loginu i hasła
* Sprzedaż biletów
* Zarządzanie biletami - anulowanie
* Dostęp do informacji o zarezerwowanych przez klientów biletach
* Dostęp do informacji o sprzedanych biletach

2.2.3. Dla kierownika

* Możliwości pracownika
* Dodawanie filmów do repertuaru kina
* Usuwanie filmów z repertuaru kina
* Dodawanie seansów z repertuaru
* Zarządzanie kontami pracowników

2.3. Wymagania niefunkcjonalne

2.3.1. Wykorzystywane technologie i narzędzia

Program do rezerwacji biletów do kina, będzie działał w sieci na serwerze Apache. System operacyjny, który będzie obsługiwać to Windows 10. Do implementacji bazy danych zostanie użyte środowisko MySQL, a do programowania język PHP. Baza danych będzie bazą relacyjną. Na początek zostanie dodanych do bazy 20 użytkowników.

2.3.2. Wymagania dotyczące rozmiaru bazy danych

* Dwie lokalizacje kina
* 2 sale kinowe w obu lokalizacjach
* Ok. 100 miejsc w każdej z sal
* Średnio 20 pracowników na kino
* Maksymalnie 30 filmów w repertuarze każdego z kin dziennie
* Seanse od 10:00 do 22:00 przez 7 dni w tygodniu
* Średnio 4 seanse dziennie w każdej z sal kinowych
* Maksymalnie 1200 sprzedanych biletów dziennie
* Ok. 5 rodzajów biletów

2.3.3. Wymagania dotyczące bezpieczeństwa systemu

* Nie będzie możliwości rejestracji przez klientów
* Możliwość logowania do systemu będą mieli tylko pracownicy kina (login+hasło)
* Szyfrowanie haseł pracowników

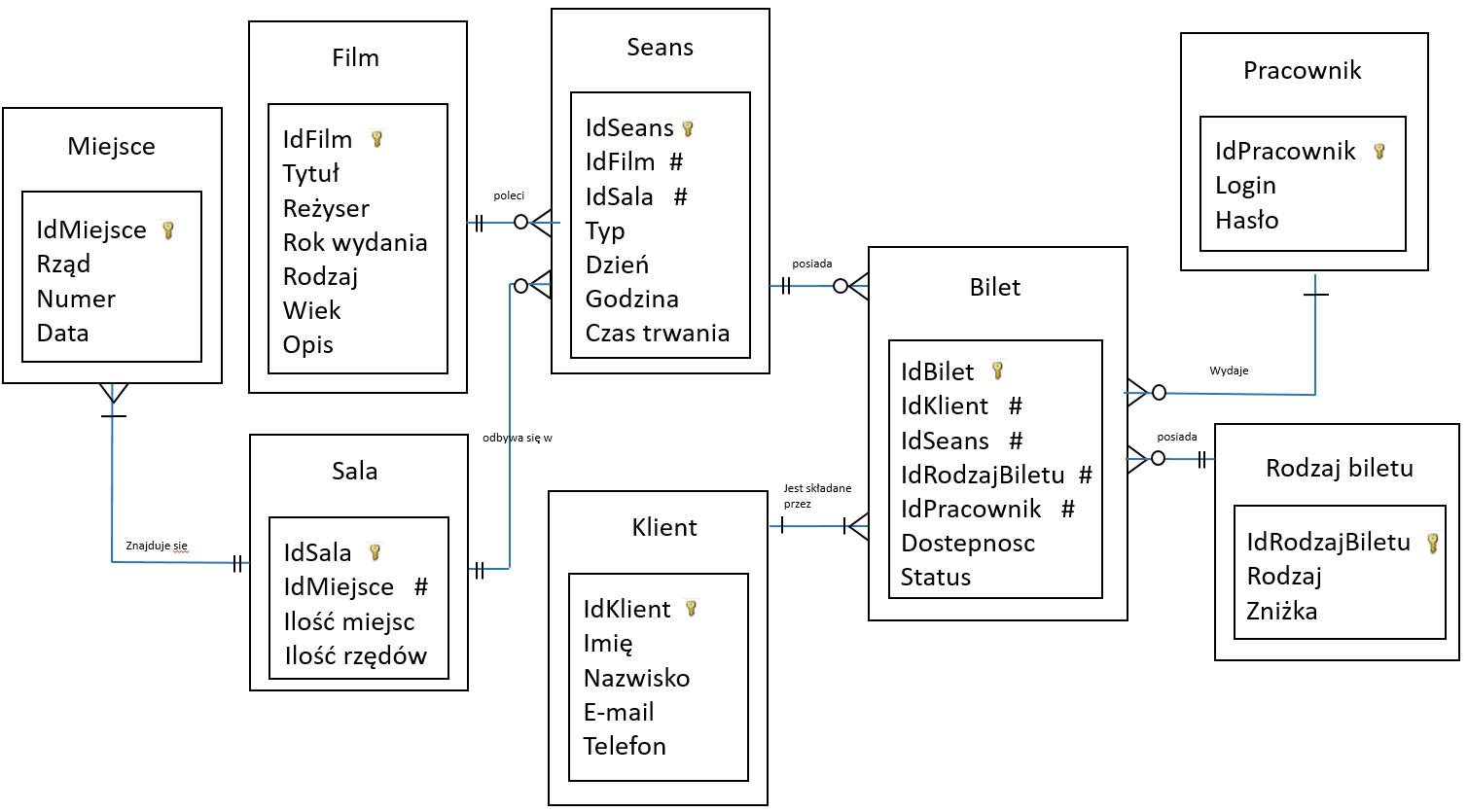
2.4. Przyjęte założenia projektowe

* Stworzenie wygodnego systemu do zakupu lub rezerwacji biletów kinowych.
* Stworzenie systemu do zarządzania siecią kin.
* System z możliwością zalogowania się przez pracowników kina.

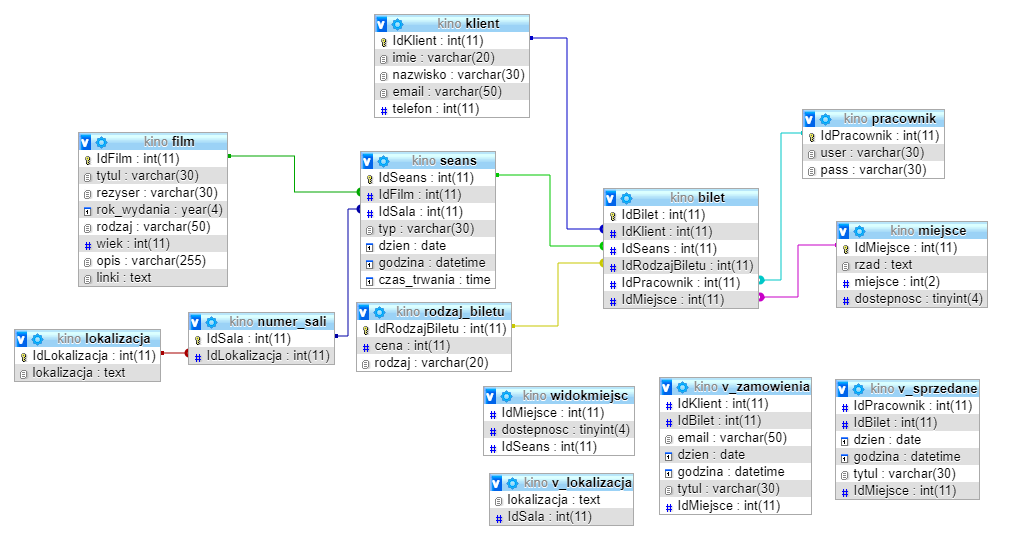
3. Projekt systemu

3.1. Projekt bazy danych

3.1.1. Model konceptualny bazy danych



3.1.2. Model fizyczny oraz logiczny bazy danych



### 

### **Tab. 1 Tabela Bilet**



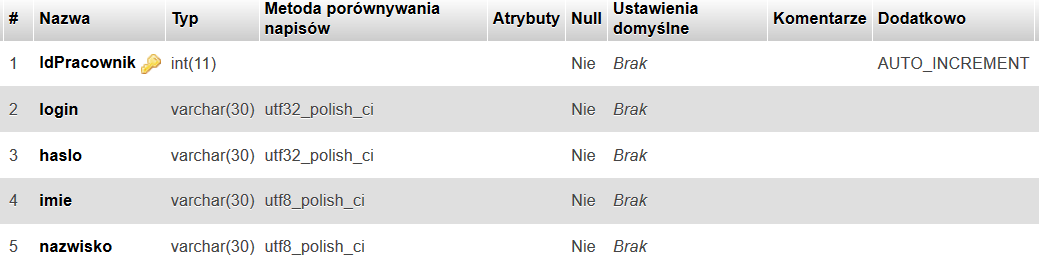
**Tab. 2 Tabela Film**



**Tab. 3 Tabela Klient**



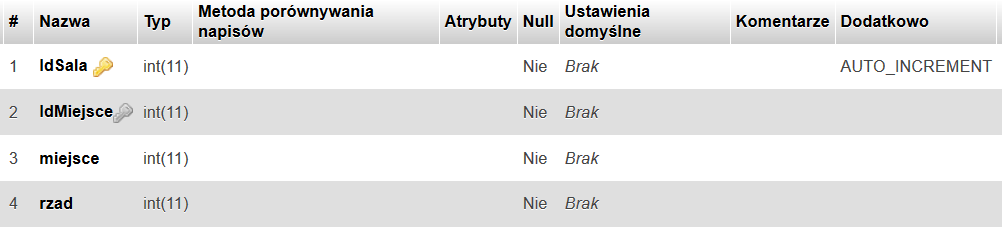
**Tab. 4 Tabela Miejsce**



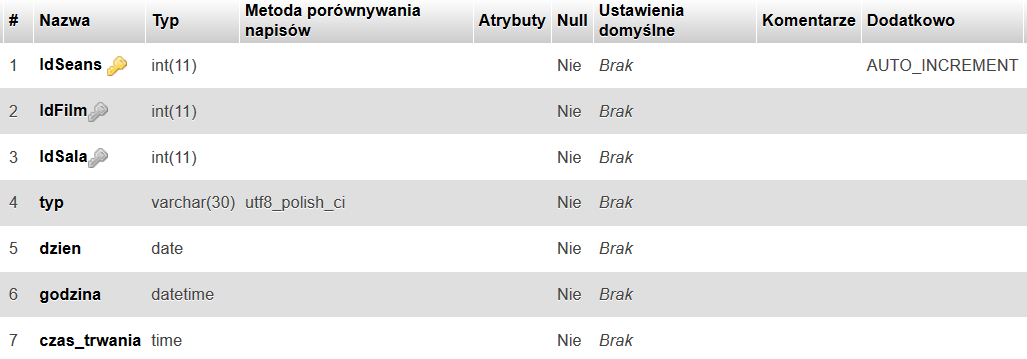
**Tab. 5 Tabela Pracownik**



**Tab. 6 Tabela Rodzaj biletu**



**Tab. 7 Tabela Sala**



**Tab. 8 Tabela Seans**

### 

3.1.3. Inne elementy schematu – mechanizmy przetwarzania danych

* Indeks złożony występujący na dwóch kolumnach w tabeli Klient, pozwoli na szybkie wyszukiwanie konkretnej osoby oraz podglądu jego rezerwacji, czy zamówienia (imię + nazwisko)
* Poprzez indeks tytuł w tabeli Film lub godzina w tabeli Seans będziemy mogli wyszukać interesujący nas film, czy sprawdzić jakie seanse są dostępne w danej godzinie
* Indeks wiek z tabeli Film pozwoli na wyszukiwanie filmów z bieżącego repertuaru odpowiednich dla wybranej grupy wiekowej.
* Indeks rodzaj z tabeli Film pozwoli na wyszukiwanie filmów z bieżącego repertuaru wybranego gatunku filmu.
* Widok *Aktualny repertuar*:

tytul (Film), godzina (Seans)

* Widok *Rodzaje biletów*:

imie (Klient), nazwisko (Klient), rodzajbiletu (Rodzaj Biletu)

* Widok *Miejsca*:

idSala (Sala), miejsce (Miejsce), rząd (Miejsce), dostepnosc (Bilet)

* Widok *Kupna/Rezerwacji*

idBiletu (Bilet), status (Bilet)

* Widok *Gatunek filmu:*

Tytuł(Film), Rodzaj(Film)

* Widok *Seanse w sali:*

idSala(Seans), Dzień(Seans),Godzina(Seans)

* Widok *v\_zamowienia:*

IdKlient(Klient), IdBilet(Bilet), email(Klient), dzien(Seans), godzina(Seans), tytul(Film), IdMiejsce(Miejsce)

* Widok *v\_sprzedane:*

IdPracownik(Pracownik), dzien(Seasn), godzina(Seans), tytul(Filmt), IdMiejsce(Miejsce)

* Widok *v\_lokalizacja:*

lokalizacja(Lokalizacja), IdSala(Sala),

* W przypadku usunięcia z harmonogramu konkretnego seansu, do Klientów, którzy wykupili bilet lub zarezerwowali miejsce będzie wysyłane powiadomienie o zaistniałym fakcie
* Po finalizacji zakupu/rezerwacji biletu przez klienta, będzie wysyłane mailowe powiadomienie o udanym zakupie/udanej rezerwacji wraz z informacją o numerze biletu.

### 

3.1.4. Projekt mechanizmów bezpieczeństwa na poziomie bazy danych

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Klient** | **Pracownik** | **Administrator** |
| **Film** | **R** | **R** | **CRUD** |
| **Seans** | **R** | **R** | **CRUD** |
| **Sala** | **R** | **R** | **CRUD** |
| **Klient** | **CRU** | **R** | **CRUD** |
| **Bilet** | **CRU** | **CRUD** | **CRUD** |
| **Miejsce** | **RU** | **RU** | **CRUD** |
| **Rodzaj biletu** | **R** | **R** | **CRUD** |
| **Pracownik** | **-** | **RU** | **CRUD** |

C- create utwórz

R-read odczytaj

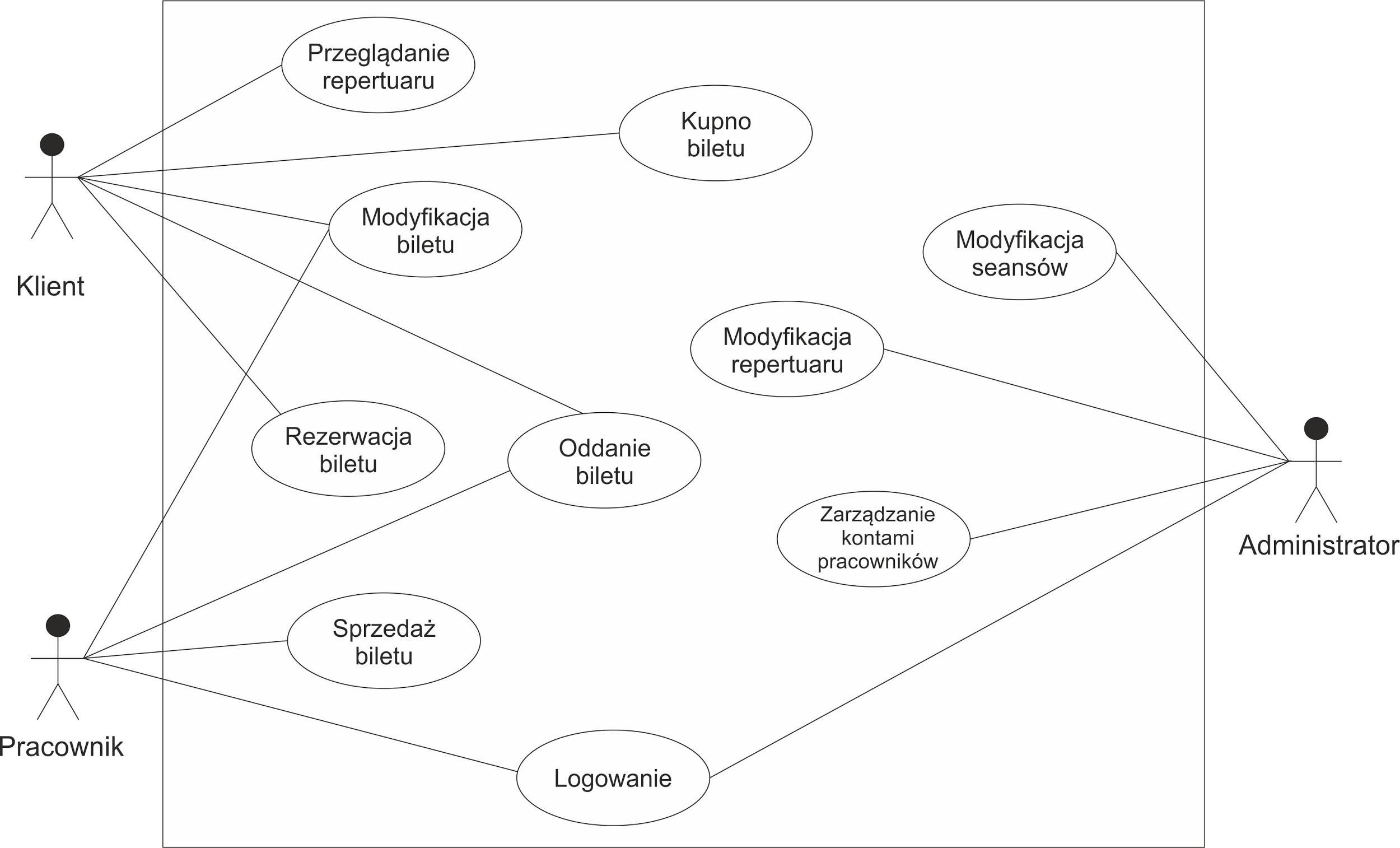
U-update aktualizuj

D-delete usuń

**Tab. 9 Uprawnienia użytkowników do poszczególnych tabel z bazy danych kina**

3.2. Projekt aplikacji użytkownika

3.2.1. Architektura aplikacji i diagramy projektowe



**Rys. 3 Diagram przypadków użycia**

3.2.2. Interfejs graficzny i struktura menu

Na głównej stronie będzie menu, w którym będą dostępne następujące opcje do wyboru:

REPERTUAR - wyświetlenie aktualnego repertuaru na najbliższy tydzień oraz możliwość rezerwacji biletów na wybrany seans

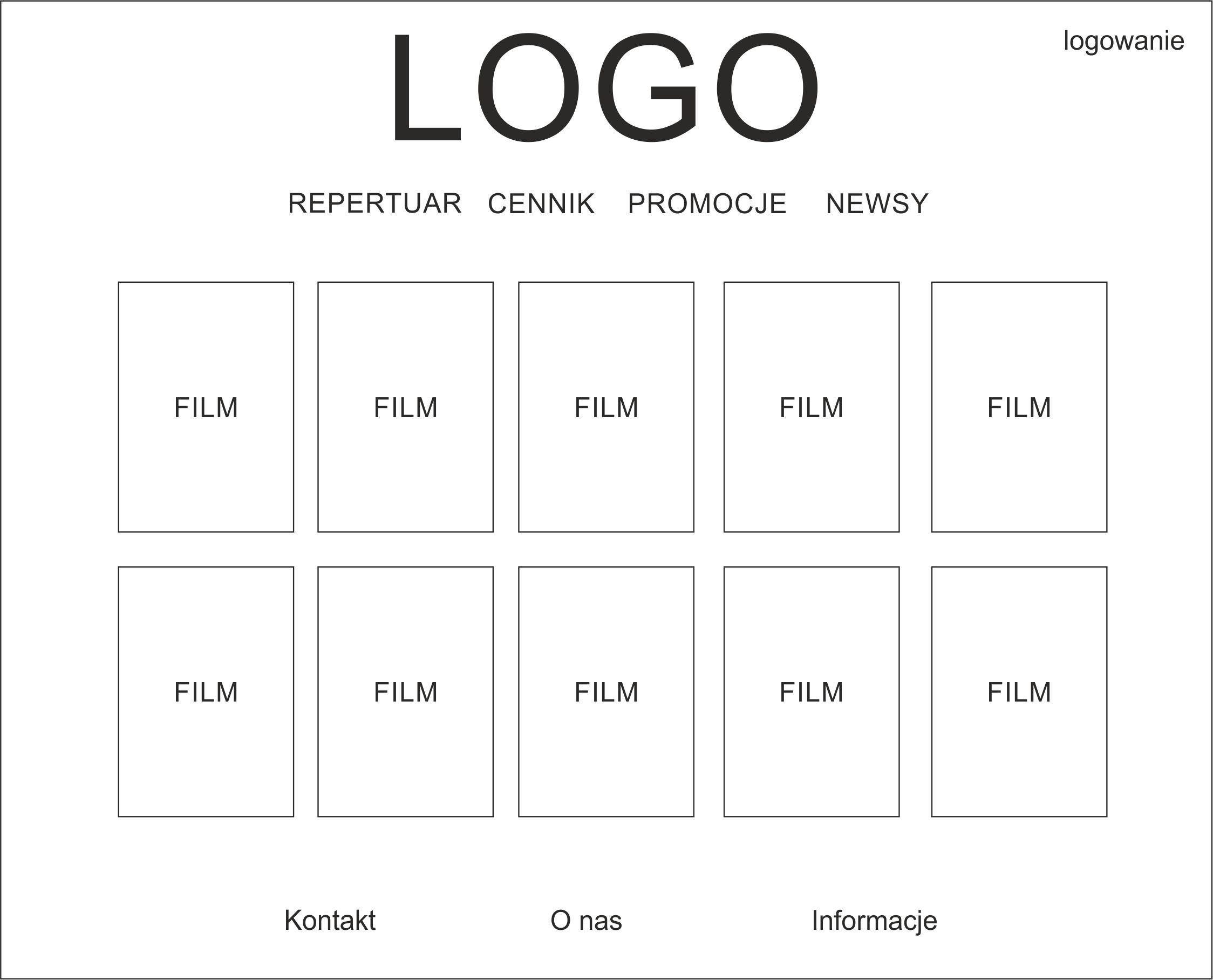
PROMOCJE - wyświetlenie listy aktualnych ofert, rabatów, pakietów dotyczących biletów kinowych

CENNIK - aktualne ceny biletów na seanse

NEWS - bieżące wiadomości o sieci kina, o premierach filmowych, komunikaty od administratora

Pod menu nawigacji będą widoczne tytuły aktualnych hitów kinowych wraz z krótkim opisem fabuły każdego filmu i najważniejszymi informacjami, które wyświetlą się po najechaniu kursorem na wybraną okładkę filmu.

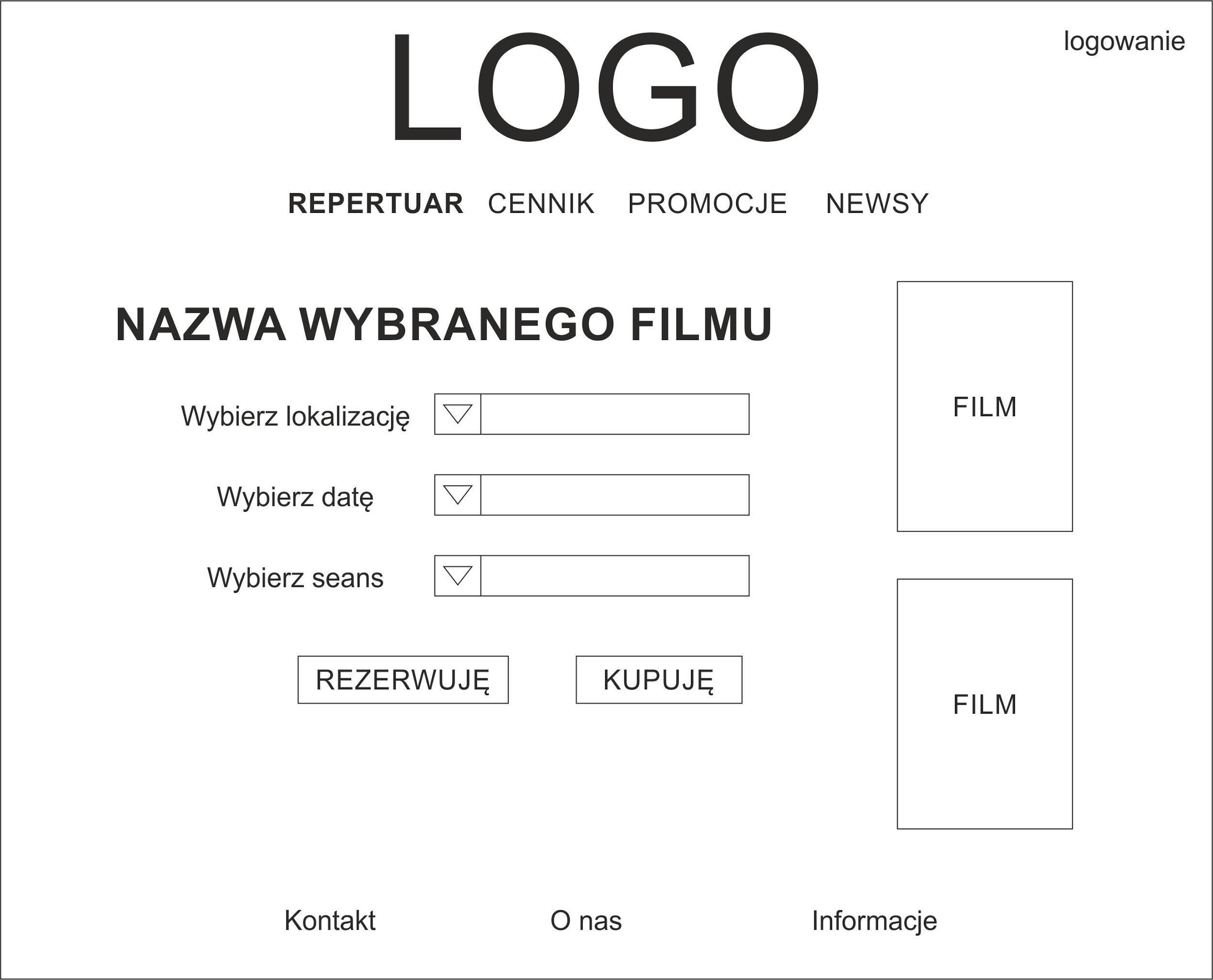
W stopce aplikacji będą informacje o kinie oraz dane kontaktowe.



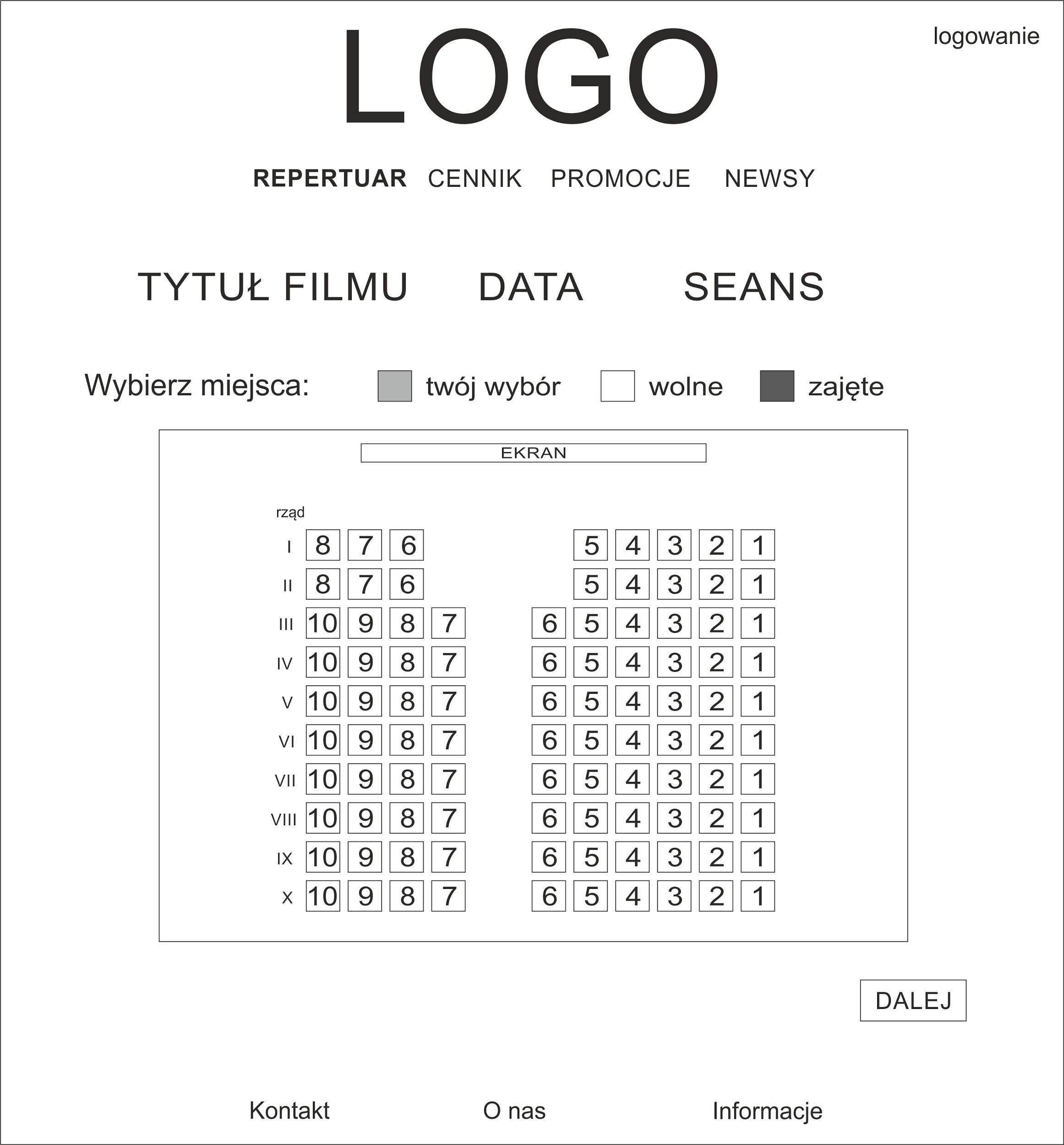
**Rys. 4 Poglądowy rozkład strony głównej aplikacji**

****

**Rys. 5 Poglądowy wygląd strony logowania dla pracowników**



**Rys. 6 Poglądowy wygląd strony wyboru seansu do rezerwacji/kupna biletu**

****

**Rys. 7 Poglądowy wygląd wyboru miejsc na seans**

3.2.3. Projekt wybranych funkcji systemu

* Funkcja rezerwacji/kupna biletu przez klienta:

Po wybraniu odpowiedniego filmu z danego w aplikacji repertuaru, wyświetli się okno z możliwością wyboru daty i godziny seansu, a następnie okno z wykazem miejsc w sali, w której wybrany seans się odbywa. Klient może wybrać jedno lub kilka miejsc w sali poprzez zaznaczenie wybranych miejsc. Klient wybiera również rodzaje biletów przypisanych do danego miejsca (normalny/ulgowy itd.) Następnie pojawiają się opcje do wyboru: rezerwacja/kupno. Aplikacja przechodzi do formularza danych klienta, w której wymagane jest podanie m.in. adresu email oraz numeru telefonu, a następnie, w zależności od wybranej opcji: rezerwacja/kupno wyświetla się odpowiednio komunikat o udanej rezerwacji/płatności.

* Funkcja sprzedaży biletów przez pracownika:

Ze strony głównej po wybraniu opcji “logowanie” wyświetla się okno jak w rysunku poglądowym nr 4. Po udanym zalogowaniu system udostępnia funkcję sprzedaży biletów na seanse. Pracownik ma możliwość rezerwacji biletów dla klientów oraz ich sprzedaży. Ma również dostęp do wszystkich

zamówień klientów. Pracownik może modyfikować zamówienia - anulować, zmienić rodzaj biletu itd. poprzez wybranie odpowiedniego rekordu z listy i za pomocą opcji “usuń” lub “zmień” może wykonać odpowiednią operację.

* Funkcja modyfikacji repertuaru dla administratora

Po zalogowaniu do systemu administrator będzie miał możliwość dodawania i usuwania filmów w repertuarze kina lub zmiany informacji dotyczących filmów. Będzie możliwość zaznaczenia wybranego filmu i za pomocą opcji “zmień” lub “usuń” będzie można wykonać odpowiednią modyfikację repertuaru. Opcja “dodaj” pozwala na dodanie do repertuaru filmu wraz z odpowiednimi informacjami (np. opis fabuły, minimalny wiek widza, gatunek itp.).

3.2.4. Metoda podłączania do bazy danych – integracja z bazą danych

Nawiązanie połączenia z bazą danych odbywa się poprzez utworzenie instancji klasy MySQLi (rozszerzenie PHP). Do konstruktora należy przekazać cztery wartości: nazwę hosta, nazwę użytkownika, hasło użytkownika i nazwę bazy danych, której będziemy używali.

3.2.5. Projekt zabezpieczeń na poziomie aplikacji

* Brak możliwości rejestracji nowych użytkowników (pracowników) przez aplikację
* Implementacja zabezpieczenia Captcha przy wypełnianiu formularza
* Odpowiednie określenie indywidualnych dla każdej grupy zasobów, do jakich będą mieli dostęp klienci, pracownicy i administrator w systemie
* Ograniczenie uprawnień dla klientów
* Przy wypełnianiu formularza przez klienta adres email musi mieć poprawną formę, a numer telefonu wyznaczoną liczbę cyfr
* Monitorowanie logowania do systemu oraz ich nieudane próby
* Wybór miejsca jest możliwy tylko wtedy, gdy dane miejsce nie jest zajęte
* Ograniczenia długości wyrazów i poszczególnych znaków wpisywanych w formularze
* Tylko administrator może dodać pracownika
* Zabezpieczenia przed niepożądanym otwarciem strony wymagające uwierzytelnienia
* Recaptcha przy dodawaniu klienta do bazy

4. Implementacja systemu baz danych

Implementacja i testy bazy danych w wybranym systemie zarządzania bazą danych.

4.1. Tworzenie tabel i definiowanie ograniczeń

Stworzono tabele jak w punkcie 3.1.2 (Tab.1 - Tab.8).

Stworzenie tabeli *bilet*:

CREATE TABLE `bilet` (

`IdBilet` int(11) NOT NULL,

`IdKlient` int(11) NOT NULL,

`IdSeans` int(11) NOT NULL,

`IdRodzajBiletu` int(11) NOT NULL,

`IdPracownik` int(11) NOT NULL,

`dostepnosc` tinyint(1) NOT NULL,

`status` tinyint(1) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf32 COLLATE=utf32\_polish\_ci;

Stworzenie tabeli *seans*:

CREATE TABLE `seans` (

`IdSeans` int(11) NOT NULL,

`IdFilm` int(11) NOT NULL,

`IdSala` int(11) NOT NULL,

`typ` varchar(30) COLLATE utf8\_polish\_ci NOT NULL,

`dzien` date NOT NULL,

`godzina` datetime NOT NULL,

`czas\_trwania` time NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_polish\_ci;

Indeksy dla tabeli *bilet*:

ALTER TABLE `bilet`

ADD PRIMARY KEY (`IdBilet`),

ADD UNIQUE KEY `klient\_id` (`IdKlient`),

ADD UNIQUE KEY `IdSeans` (`IdRodzajBiletu`,`IdPracownik`),

ADD KEY `wydaje` (`IdPracownik`),

ADD KEY `ma` (`IdSeans`);

AUTO\_INCREMENT dla tabeli *bilet*:

ALTER TABLE `bilet`

MODIFY `IdBilet` int(11) NOT NULL AUTO\_INCREMENT;

Ograniczenia dla tabeli *seans*:

ALTER TABLE `seans`

ADD CONSTRAINT `odbywa sie w` FOREIGN KEY (`IdSala`) REFERENCES `sala` (`IdSala`),

ADD CONSTRAINT `poleci` FOREIGN KEY (`IdFilm`) REFERENCES `film` (`IdFilm`) ON UPDATE CASCADE;

4.2. Implementacja mechanizmów przetwarzania danych

Stworzenie indeksów:

* CREATE INDEX indeks\_imie\_nazw ON klient (imie, nazwisko)
* CREATE INDEX indeks\_tytul ON film (tytul)
* CREATE INDEX indeks\_godzina\_seansu ON seans (godzina)
* CREATE INDEX indeks\_ogr\_wieku ON film (wiek)
* CREATE INDEX indeks\_gatunek ON film (rodzaj)

Implementacja widoków (przykłady) :

* Widok pokazujący lokalizację sal:

CREATE VIEW v\_lokalizacja AS

SELECT l.lokalizacja, ns.IdSala FROM numer\_sali ns

LEFT JOIN lokalizacja l on l.IdLokalizacja=ns.IdLokalizacja

* Widok wyszukujący gatunek filmu:

CREATE VIEW Gatunek filmu

AS

SELECT tytul, rodzaj

FROM film;

gdzie możemy się później odwołać do widoku, wyszukując gatunek ‘Animacja’:

SELECT \*

FROM Gatunek filmu

WHERE title LIKE 'Animacja';

* Widok zamówień dokonanych przez klientów:

CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW v\_zamowienia AS select k.`IdKlient` AS IdKlient,`b`.`IdBilet` AS IdBilet,`k`.`email` AS email,`s`.`dzien` AS dzien,`s`.`godzina` AS godzina,`f`.`tytul` AS tytul,`m`.`IdMiejsce` AS IdMiejsce from ((((klient k left join bilet b on((b.`IdKlient` = k.`IdKlient`))) left join seans s on((s.`IdSeans` = b.`IdSeans`))) left join film f on((f.`IdFilm` = s.`IdFilm`))) left join miejsce m on((m.`IdMiejsce` = b.`IdMiejsce`))) ;

* Widok zamówień dokonanych przez pracowników został stworzony analogicznie do powyższego.

4.3. Implementacja uprawnień i innych zabezpieczeń

Na podstawie tabeli Tab.9 nadano uprawnienia użytkownikom:

Nadanie wszystkich uprawnień w całej bazie danych *kino* dla administratora:

GRANT ALL PRIVILEGES ON `kino`.\* TO 'administrator'@'localhost';

Przykładowe uprawnienia do tabel *bilet* i *miejsce* dla pracowników

GRANT ALL PRIVILEGES ON bilet TO ‘pracownik’@’localhost’;

GRANT SELECT, UPDATE ON miejsce TO ‘pracownik’@’localhost’;

Przykładowe uprawnienia do tabeli *bilet* dla klientów:

GRANT SELECT, UPDATE ON bilet TO ‘klient’@’localhost’;

4.4. Testowanie bazy danych na przykładowych danych

* Dodawanie rekordów do bazy:

Dodano przykładowe rekordy do wybranych tabel, np. w tabeli *film* dodano 7 filmów.

Przykład dodawania filmu:

INSERT INTO `film` (`IdFilm`, `tytul`, `rezyser`, `rok\_wydania`, `rodzaj`, `wiek`, `opis`) VALUES

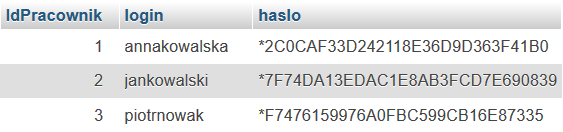
(1, 'Kształt wody', 'Guillermo del Toro', 2017, 'Fantasy', 12, 'Elisa Esposito pracuje jako woźna w tajnym rządowym laboratorium. Jej życie ulega nagłej zmianie, gdy odkrywa pilnie strzeżony sekret.'),



**Tab. 10 Tabela *film* z bazy *kino***

* Wyświetlanie widoku tabeli pracownik

SELECT \* FROM pracownik;



**Tab. 11 Tabela *pracownik* z bazy *kino***

Dodawanie wartości do tabel przebiega prawidłowo. Dane są poprawnie odczytywane wraz z napisami w języku polskim.

Gdy kolumna jest oznaczona jako NOT NULL w trakcie zapisu występuje błąd, który należy poprawić, bądź do tabeli są wpisywane 4 zera (0000).

Hasło pracownika jest zmienną Password, więc jest zaszyfrowane.

* Wybieranie rekordów z bazy

Przykład wybrania wartości z tabeli *film*:

SELECT tytul, rezyser, wiek FROM film;



**Tab. 12 Wybrane rekordy z bazy**

SELECT \* FROM film WHERE rodzaj = 'sensacyjny'



**Tab. 13 Wynik zapytania SELECT**

5. Implementacja i testy aplikacji

5.1. Instalacja i konfigurowanie systemu

Aby aplikacja działała poprawnie należy pobrać oraz zainstalować na komputerze serwer lokalny xampp. Po uruchomieniu panelu kontrolnego xampp należy załączyć moduł: Apache oraz MySQL. W przeglądarkę internetową należy wpisać localhost/phpmyadmin, następnie utworzyć nową bazę danych o nazwie kino i wgrać bazę danych aplikacji kino.sql. Wszystkie pliki powinny znajdować się w folderze htdocs. Aby się tam dostać, trzeba wejść w dysk, na którym został zainstalowany xampp, następnie w folder xampp, oraz htdocs (przykładowa ścieżka do folderu: C:\xampp\htdocs) Gdy to już zostanie wykonane, w nowym oknie przeglądarki wpisać adres localhost/kino/mem.php. Aplikacja została poprawnie uruchomiona.

Aby wgrać pliki na serwer ftp potrzebny jest program, który to umożliwi oraz dane do serwera ftp (adres hosta, login i hasło). W programie (np. FileZilla) należy połączyć się z serwerem podając adres serwera, login oraz hasło, a po udanym połączeniu “przenieść” wybrane pliki (\*.php oraz kino.sql)  z okna programu z zawartością dysku do wybranego katalogu z okna z zawartością serwera ftp. Po udanym przesłaniu plików można wylogować się z serwera i zamknąć program. Aplikacja powinna poprawnie działać na wybranym serwerze.

5.2. Instrukcja użytkowania aplikacji

**5.2.1. Instrukcja obsługi zakupu biletu dla klienta**

1. Otworzyć plik mem.php
2. Wejść w repertuar
3. Wybrać film
4. Wybrać dzień oraz godzinę seansu w wybranej lokalizacji
5. Wybrać miejsca
6. Wybrać ilość i rodzaje biletów
7. Wypełnić formularz danymi klienta

**5.2.2. Instrukcja obsługi sprzedaży biletu dla pracownika**

1. Otworzyć plik mem.php
2. Zalogować się
3. Wybrać film
4. Wybrać dzień oraz godzinę seansu
5. Wybrać miejsce
6. Wybrać ilość i rodzaje biletów
7. Zatwierdzić wybór

**5.2.3. Instrukcja obsługi dodawania filmu dla pracownika**

1. Otworzyć plik mem.php
2. Zalogować się
3. Wejść w dodawanie filmu
4. Uzupełnić dane filmu

**5.2.4. Instrukcja obsługi dodawania seansu dla pracownika**

1. Otworzyć plik mem.php
2. Zalogować się
3. Wejść w dodawanie seansu
4. Uzupełnić dane seansu

5.3. Testowanie opracowanych funkcji systemu

### **5.3.1. Przejście do stron niedostępnych dla klienta**

Wpisując nazwy stron w przeglądarkę, które są dostępne po uwierzytelnieniu tylko dla pracowników aplikacja przenosi nas do strony logowanie.php, gdzie należy podać dane pracownika, aby przejść dalej.

### **5.3.2. Próba zarezerwowania zajętego miejsca**

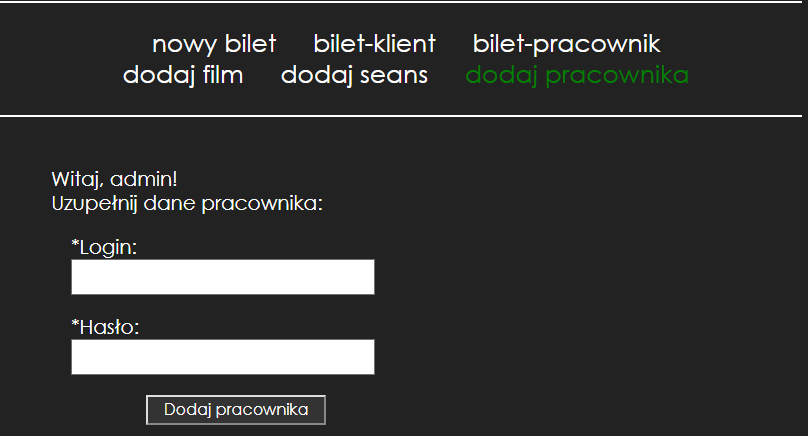
Miejsce, które zostało już zajęte zamiast checkboxa wyświetla ikonę która nie jest interaktywna - brak możliwości ponownego wybrania tego miejsca.



**Rys. 8 Przykładowe miejsca wolne (checkboxy) i zarezerwowane (ikony).**

5.3.3. Dodanie pracownika przez administratora

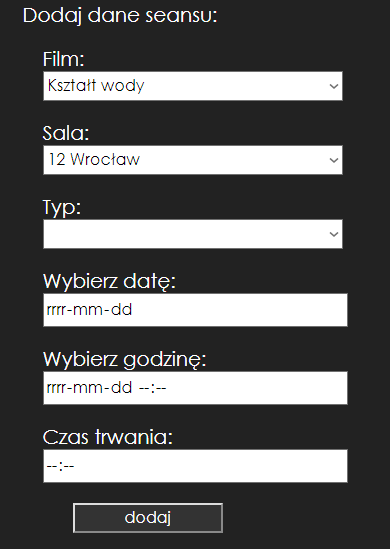
Do programu zostało dodane zabezpieczenie, które ma za zadanie sprawdzać, kto jest zalogowany, a do strony *dodaj pracownika* dać dostęp tylko administratorowi. Dla pracownika opcja ta przenosi do zakładki *nowy bilet* (strona główna po zalogowaniu).



**Rys. 9 Zrzut z aplikacji – dodawanie pracownika**

5.3.4. Dodawanie seansu

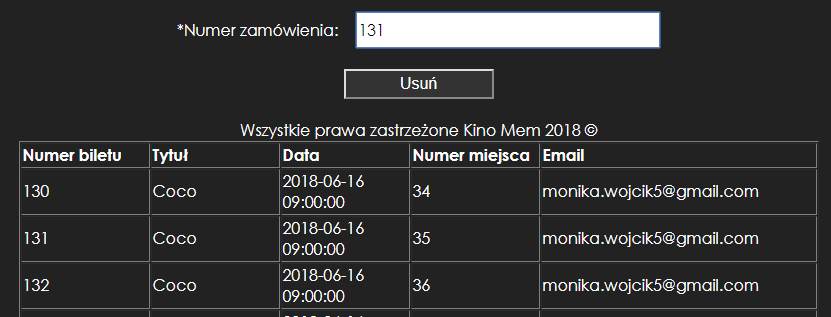
Pracownik oraz administrator ma możliwość dodania seansu. Aby dodać seans należy wejść na stronę *dodanie seansu*, gdzie pojawi się formularz. Aby dodawanie było łatwiejsze, dane zostały pobrane z bazy oraz przedstawione za pomocą listy rozwijanej. Typ pól do wpisywania danych zgadza się z typem danych w bazie.



**Rys. 10 Zrzut z aplikacji – dodawanie seansu**

5.3.5. Anulowanie zamówienia

Na stronie *baza-klient* dostępnej tylko po uwierzytelnieniu jest wyświetlona baza wszystkich biletów zakupionych przez klienta. Pracownik ma możliwość anulowania biletu. Aby anulować bilet należy wpisać odpowiedni numer biletu, który chcemy usunąć oraz zatwierdzić wybór. Bilet automatycznie usuwa się z bazy.



**Rys.11 Zrzut z aplikacji – usuwanie zamówienia (widok przed usunięciem)**

### 

**Rys.12 Zrzut z aplikacji – usuwanie zamówienia (widok po usunięciu)**

5.4. Omówienie wybranych rozwiązań programistycznych

5.4.1. Implementacja interfejsu dostępu do bazy danych

Aby nawiązać połączenie z bazą danych należy skorzystać z rozszerzenie PHP o nazwie MySQLi.

Listing 1: Nawiązanie połączenia z bazą danych kino

https://lh3.googleusercontent.com/UvsWK1S2qXu-Gd9iYK8YXdxg6vwRJp0Rw448y9sxBfTQsjhrxwJPpvuqfUlvALve8Bo5SkFHMhc7LN5SSJkMVjHSheDo2Gfd1WRdvTgPnVcT9vYxFmWHRBvLaWSRiqALsPH3gfPT

Na powyższym listingu “localhost” jest nazwą serwera mysql, “root” to login administratora bazy danych, kolejnym elementem jest hasło (tutaj brak), natomiast “kino” jest nazwą bazy danych, z którą nawiązujemy połączenie.

Aby odczytać wybrane dane z bazy, z którą nawiązano połączenie, należy podać odpowiednie zapytanie SQL. Przykładowe zapytania:

Listing 2: Zapytanie SQL w kodzie aplikacji - cena biletu normalnego

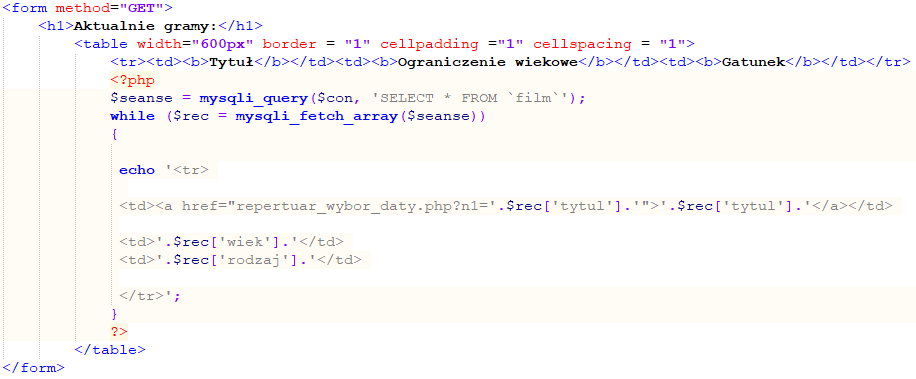
https://lh6.googleusercontent.com/avzsYRYFnFkyxm017CFcIIRXXEK4WT6kaO-MuXrZA4YcRd6kUJF79pajURKscTlkHIs3Bf4m0OLYULjtQPHEqqj3LTixahd6med9CoKOTVI81zEpBsXP7uNZ5ccH7LQBKhPSvYCr

Listing 3: Zapytanie SQL w kodzie aplikacji - dodanie danych klienta do bazy

https://lh3.googleusercontent.com/PbhmgIzMIXW1kohmM_oqHkofwBOw7GAEJdqvGC2mTdeZ9-RHATqdaQpP9HJ6OagpfKwpNF46E-dJzYmTqbs7hGzcSeFR19hYRPjxPn9aMay_OIodR1BJEwVVivtxvtLs13pTZluE

5.4.2. Implementacja wybranych funkcjonalności systemu

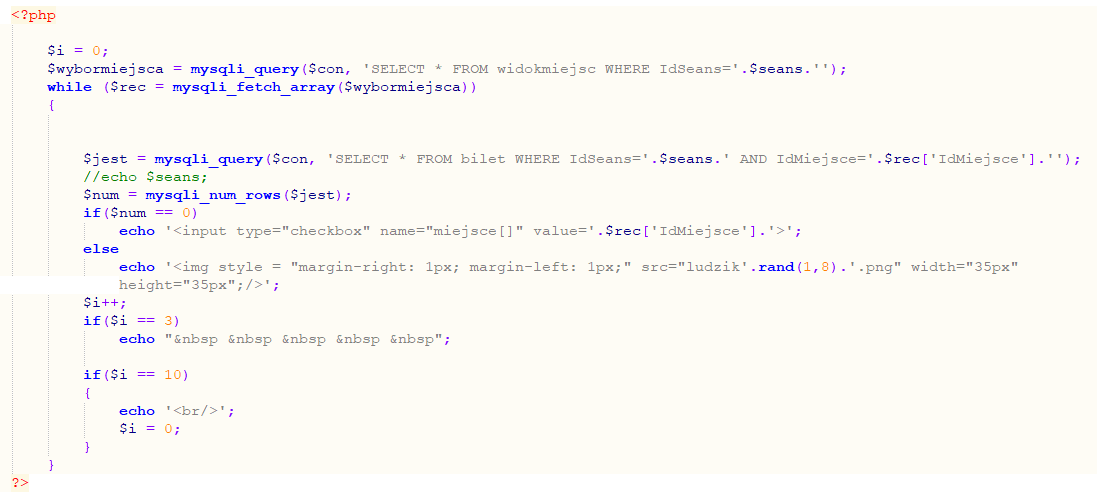
Listing 4: Pobieranie danych z bazy i wyświetlanie w tabeli – repertuar

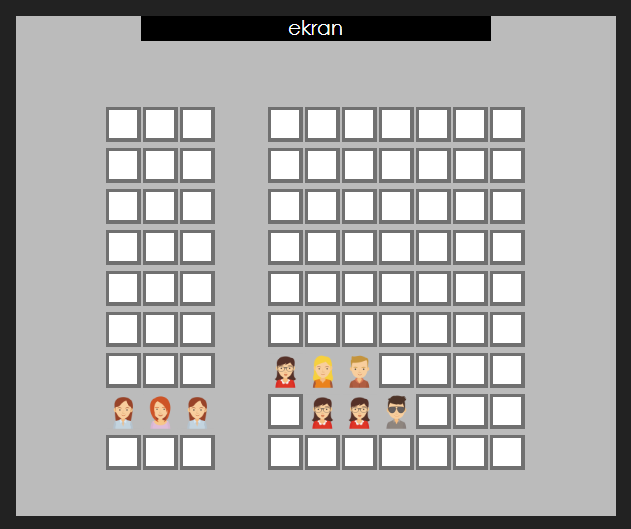




**Rys. 13 Repertuar - wynik implementacji kodu z Listingu 4**

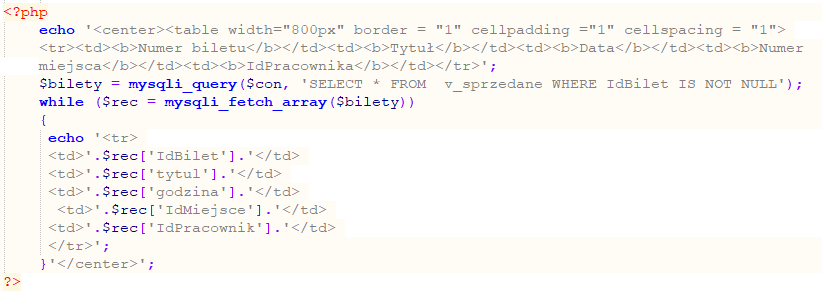
Listing 5: Pobranie danych z bazy i wyświetlenie - miejsca w sali kinowej

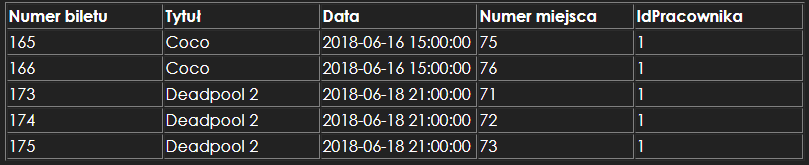




**Rys. 14 Miejsca w sali kinowej - wynik implementacji kodu z Listingu 5**

Listing 6: Pobieranie danych z widoku i wyświetlanie - bilety sprzedane





**Rys. 15 Sprzedane bilety - wynik implementacji kodu z Listingu 6**

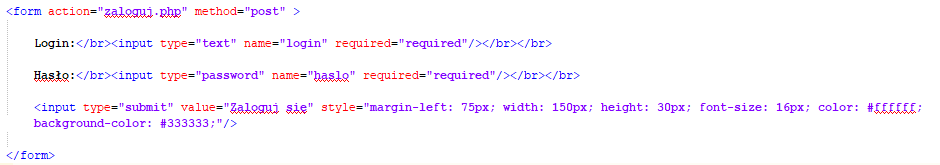
5.4.3. Implementacja mechanizmów bezpieczeństwa

* Logowanie do systemu

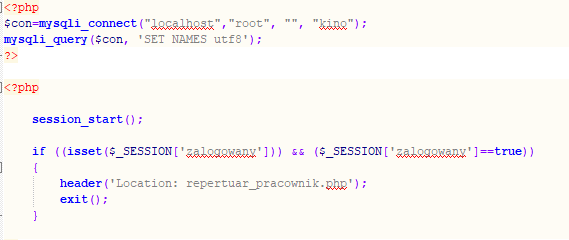
Na stronie *logowanie.php* użytkownik proszony jest o wypełnienie pól odpowiednimi danymi - loginem oraz hasłem przypisanym do konta. Po wpisaniu następuje identyfikacja użytkownika - czy użytkownik o takiej nazwie istnieje w bazie oraz uwierzytelnienie -  sprawdzenie, czy podane hasło jest prawidłowe, pasujące do podanego loginu. Jeśli nie podano hasła/loginu bądź są one błędne, zostaje wyświetlony formularz wraz z informacjami o błędach. W przypadku, gdy dane są prawidłowe, następuje autoryzacja, rozpoczyna się sesja i zostaje zarejestrowana zmienna *$zalogowany.* Następuje przekierowanie pracownika do chronionej strony *repertuar\_pracownik.php*. Funkcja *session\_start()* będzie wznawiała sesję rozpoczętą podczas logowania. Próby nieautoryzowanego dostępu do strony będą niepowodzeniem, ponieważ nie będą zawierały ustawionej zmiennej *$zalogowany*, a więc niezalogowany użytkownik zostanie przekierowany do skryptu uwierzytelniającego (formularz logowania na stronie *logowanie.php*).

Aby wylogować użytkownika z systemu należy wznowić sesję (*session\_start*), a następnie ją zakończyć (*session\_unset*). Wylogowanie następuje po naciśnięciu przycisku *wyloguj się*.

Listing 7: Implementacja formularza do logowania dla pracowników



Listing 8: Sprawdzenie, czy istnieje zmienna sesyjna



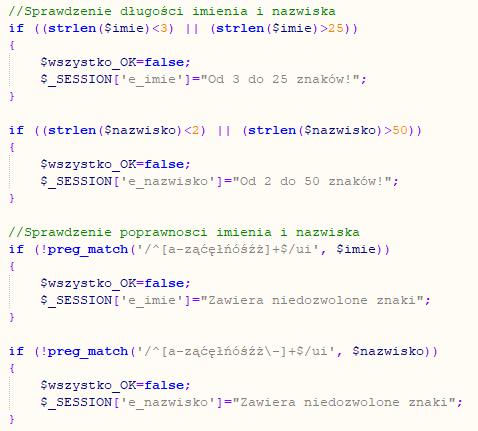
* Zabezpieczenia przy wypełnianiu formularza

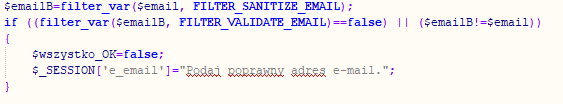
Wypełnienie formularza z danymi odbywa się po poprawnym wybraniu seansu oraz miejsc w sali. W formularzu należy podać imię, nazwisko, email oraz numer telefonu, zaakceptować regulamin i zatwierdzić zabezpieczenie Recaptcha. Wszystkie dane muszą być poprawnie wypełnione. Dane zostają przesłane do bazy do tabeli klient oraz bilet. W tabeli klient zostają wpisane dane klienta, a w tabeli bilet tworzona jest odpowiednia ilość wierszy w zależności od ilości zakupionych biletów.

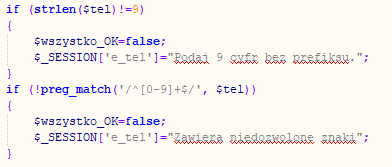


**Rys. 16 Widok ostrzeżeń przy błędnym wypełnieniu formularza (brak wprowadzonych danych)**

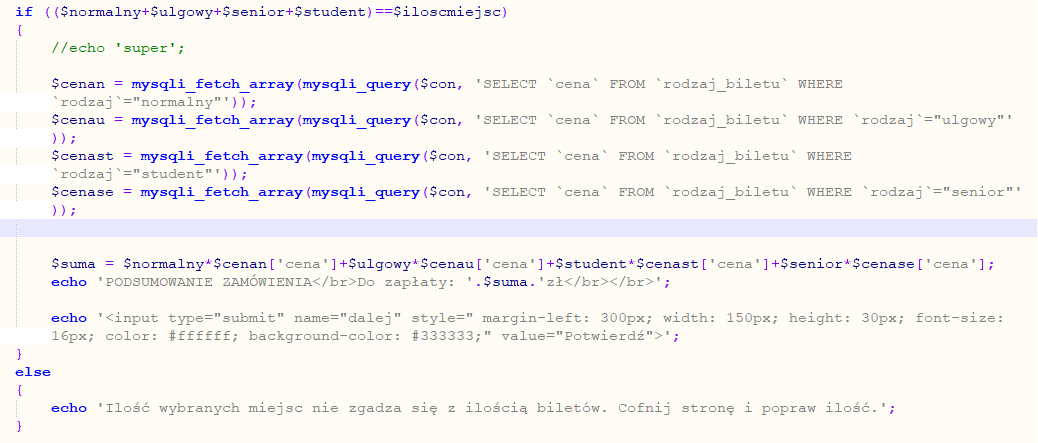
Listing 9: Zabezpieczenia poprawności wprowadzonych danych



Listing 10: Sprawdzenie poprawności email

Listing 11: Sprawdzenie poprawności numeru telefonu - musi się składać z 9 cyfr:

* Sprawdzenie, czy ilość miejsc zgadza się z ilością wybranych biletów (brak możliwości przejścia do formularza w przypadku błędu)

Listing 12: Sprawdzenie poprawności wprowadzonej ilości miejsc

6. Podsumowanie i wnioski

Projekt został wykonany zgodnie z wcześniej ustalonymi założeniami. Została stworzona aplikacja, która umożliwia w prosty sposób zarezerwowanie miejsca na wybrany seans, a także umożliwia zarządzanie kinem, np. dodawanie seansów. Zostały zaimplementowane zabezpieczenia ułatwiające łączenie aplikacji z bazą. Zaimplementowane funkcje działają poprawnie - brak błędów podczas działania (rezerwacji biletów, dodania pracownika itp.).

Zostały wykonane testy sprawdzające poprawność działania aplikacji oraz jej połączenia z bazą danych. Wynik testów jest pozytywny - projekt spełnia założenia.

Bibliografia

1. [*http://roman.ptak.staff.iiar.pwr.wroc.pl/BD\_wyklad\_nr1\_ver6.pdf*](http://roman.ptak.staff.iiar.pwr.wroc.pl/BD_wyklad_nr1_ver6.pdf)
2. [*http://it.dth.pl/create-update-alter-view-tworzenie-i-modyfikacja-widokow-kurs-jezyka-sql/*](http://it.dth.pl/create-update-alter-view-tworzenie-i-modyfikacja-widokow-kurs-jezyka-sql/)
3. [*https://stackoverflow.com*](https://stackoverflow.com)
4. Duckett Jon *“HTML i CSS”,* HELION, 2014
5. *(Pasja Informatyki,* [*Kurs PHP. Programowanie backendowe)*](https://www.youtube.com/watch?v=WSeKPbVZBoo&list=PLOYHgt8dIdox81dbm1JWXQbm2geG1V2uh) *(*[*https://youtu.be/tD0Q5QwoQJI*](https://youtu.be/tD0Q5QwoQJI)*)*

Spis rysunków

Rys. 1 Rozkład kina w jednej z lokalizacji

Rys. 2 Przykładowy rozkład miejsc w sali kinowej

Rys. 3 Diagram przypadków użycia

Rys. 4 Poglądowy rozkład strony głównej aplikacji

Rys. 5 Poglądowy wygląd strony logowania dla pracowników

Rys. 6 Poglądowy wygląd strony wyboru seansu do rezerwacji/kupna biletu

Rys. 7 Poglądowy wygląd wyboru miejsc na seans

Rys. 8 Zrzut z aplikacji – widok zajętych miejsc

Rys. 9 Zrzut z aplikacji – dodawanie pracownika

Rys. 10 Zrzut z aplikacji – dodawanie seansu

Rys. 11 Zrzut z aplikacji – usuwanie zamówienia (widok przed usunięciem)

Rys. 12 Zrzut z aplikacji – usuwanie zamówienia (widok po usunięciu)

Rys. 13 Repertuar - wynik implementacji kodu z Listingu 4

Rys. 14 Miejsca w sali kinowej - wynik implementacji kodu z Listingu 5

Rys. 15 Sprzedane bilety - wynik implementacji kodu z Listingu 6

Rys. 16 Widok ostrzeżeń przy błędnym wypełnieniu formularza (brak wprowadzonych danych)

Spis tabel

Tab. 1 Tabela Bilet

Tab. 2 Tabela Film

Tab. 3 Tabela Klient

Tab. 4 Tabela Miejsce

Tab. 5 Tabela Pracownik

Tab. 6 Tabela Rodzaj biletu

Tab. 7 Tabela Sala

Tab. 8 Tabela Seans

Tab. 9 Uprawnienia użytkowników do poszczególnych tabel z bazy danych kina

Tab. 10 Tabela film z bazy kino

Tab. 11 Tabela pracownik z bazy kino

Tab. 12 Wybrane rekordy z bazy

Tab. 13 Wynik zapytania SELECT

Spis listingów

Listing

Spis treści

[1. Wstęp 2](#_Toc516128304)

[1.1. Cel projektu 2](#_Toc516128305)

[1.2. Zakres projektu 2](#_Toc516128306)

[2. Analiza wymagań 2](#_Toc516128307)

[2.1. Opis działania i schemat logiczny systemu 2](#_Toc516128308)

[2.2. Wymagania funkcjonalne 4](#_Toc516128309)

[2.2.1. Dla klienta 4](#_Toc516128310)

[2.2.2. Dla pracownika 4](#_Toc516128311)

[2.2.3. Dla kierownika 4](#_Toc516128312)

[2.3. Wymagania niefunkcjonalne 4](#_Toc516128313)

[2.3.1. Wykorzystywane technologie i narzędzia 4](#_Toc516128314)

[2.3.2. Wymagania dotyczące rozmiaru bazy danych 5](#_Toc516128315)

[2.3.3. Wymagania dotyczące bezpieczeństwa systemu 5](#_Toc516128316)

[2.4. Przyjęte założenia projektowe 5](#_Toc516128317)

[3. Projekt systemu 6](#_Toc516128318)

[3.1. Projekt bazy danych 6](#_Toc516128319)

[3.1.1. Model konceptualny bazy danych 6](#_Toc516128320)

[3.1.2. Model fizyczny oraz logiczny bazy danych 6](#_Toc516128321)

[3.1.3. Inne elementy schematu – mechanizmy przetwarzania danych 9](#_Toc516128323)

[3.1.4. Projekt mechanizmów bezpieczeństwa na poziomie bazy danych 10](#_Toc516128324)

[3.2. Projekt aplikacji użytkownika 11](#_Toc516128360)

[3.2.1. Architektura aplikacji i diagramy projektowe 11](#_Toc516128361)

[3.2.2. Interfejs graficzny i struktura menu 11](#_Toc516128362)

[3.2.3. Projekt wybranych funkcji systemu 14](#_Toc516128363)

[3.2.4. Metoda podłączania do bazy danych – integracja z bazą danych 14](#_Toc516128364)

[3.2.5. Projekt zabezpieczeń na poziomie aplikacji 14](#_Toc516128365)

[4. Implementacja systemu baz danych 15](#_Toc516128366)

[4.1. Tworzenie tabel i definiowanie ograniczeń 15](#_Toc516128367)

[4.2. Implementacja mechanizmów przetwarzania danych 16](#_Toc516128368)

[4.3. Implementacja uprawnień i innych zabezpieczeń 16](#_Toc516128369)

[4.4. Testowanie bazy danych na przykładowych danych 17](#_Toc516128370)

[5. Implementacja i testy aplikacji 18](#_Toc516128371)

[5.1. Instalacja i konfigurowanie systemu 18](#_Toc516128372)

[5.2. Instrukcja użytkowania aplikacji 19](#_Toc516128373)

[5.3. Testowanie opracowanych funkcji systemu 19](#_Toc516128374)

[5.3.1. Przejście do stron niedostępnych dla klienta 19](#_Toc516128375)

[5.3.2. Próba zarezerwowania zajętego miejsca 19](#_Toc516128376)

[5.3.3. Dodanie pracownika przez administratora 20](#_Toc516128377)

[5.3.4. Dodawanie seansu 20](#_Toc516128378)

[5.3.5. Anulowanie zamówienia 21](#_Toc516128379)

[5.4. Omówienie wybranych rozwiązań programistycznych 22](#_Toc516128381)

[5.4.1. Implementacja interfejsu dostępu do bazy danych 22](#_Toc516128382)

[5.4.2. Implementacja wybranych funkcjonalności systemu 23](#_Toc516128383)

[5.4.3. Implementacja mechanizmów bezpieczeństwa 25](#_Toc516128384)

[6. Podsumowanie i wnioski 28](#_Toc516128385)

[Bibliografia 28](#_Toc516128386)

[Spis rysunków 28](#_Toc516128387)

[Spis tabel 29](#_Toc516128388)

[Spis listingów 29](#_Toc516128389)

Spis treści.................................................................................................................................30