

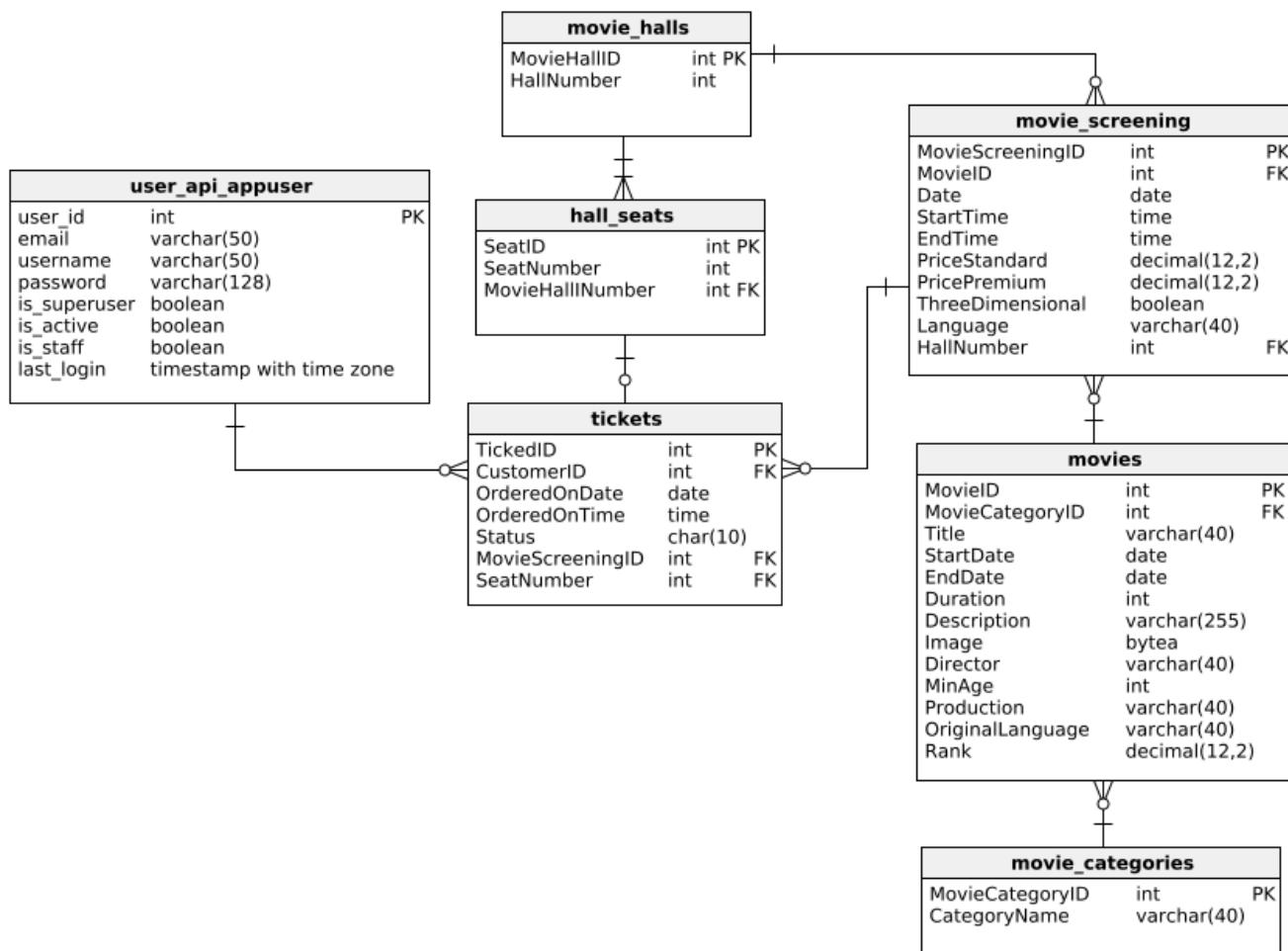
Bazy danych 2 Projekt

Stas Kochevenko, Wiktor Dybalski

1. Opis projektu

W ramach projektu została stworzona strona kina z możliwością rejestracji, logowania się na stronie, rezerwacją i zakupem biletów na dostępne seanse. Wybrane przez nas technologie: PostgreSQL (baza danych), ReactJS (frontend), Django (backend).

2. Schemat bazy danych



Dany schemat przedstawia podstawowe encje, niezbędne do poprawnego działania strony kina. Do uproszczenia modelu bazy danych przyjęliśmy kilka zasad:

1. Kino posiada tylko jedną lokalizację
2. Wszystkie filmy są przydzielone tylko jednej kategorii oraz posiadają tylko jednego reżysera
3. Seanse filmów odbywają się w salach (1-6), przy czym każda sala posiada taką samą liczbę miejsc (84), chociaż to zawsze można zmienić

4. Bilety na seans mają dwie ceny: Standard & Premium. Premium-bilet odpowiada ostatniemu rzędowi w kinie, czyli miejscom 1-11
5. Klienci mogą kupować i rezerwować bilety na określone miejsca. W przypadku rezerwacji w profilu mają oni możliwość kupienia danego biletu albo jego anulowania
6. Każdy bilet posiada status. Istnieją 3 opcje tego statusu: "New" - nowy nieopłacony bilet (rezerwacja), "Confirmed" - opłacony bilet (nie da się jego anulować), "Canceled" - anulowany bilet
7. Klienci nie mogą zarezerwować miejsca na seans, do rozpoczęcia którego pozostało mniej niż 2 godziny. W takim przypadku mogą oni wyłącznie kupić bilet na dane miejsce
8. Na stronie ustawiliśmy datę 22-05-2024 oraz czas 13-00, który jest "aktualnym" czasem przeglądania strony. To jest umotywowane tym, że filmy i seansy są ustawione na określone daty, więc stała data była wygodna do testowania poprawności działania systemu i bazy danych
9. Nie udało się zrealizować utworzenia sesji użytkownika na froncie (mimo, że logowanie i rejestracja na poziomie backendu i bazy danych działają poprawnie), więc na stronie ustawiliśmy użytkownika domyślnego i ID = 6

3. Tabele

- user_api_app_user

Tabela została wygenerowana przez django, połączymy ją z tabelą Tickets

```
create table user_api_appuser
(
    password      varchar(128) not null,
    last_login    timestamp with time zone,
    is_superuser  boolean      not null,
    user_id       integer      generated by default as identity
        primary key,
    email         varchar(50)  not null
        unique,
    username      varchar(50)  not null,
    is_active     boolean      not null,
    is_staff      boolean      not null
);

alter table user_api_appuser
    owner to postgres;

create index user_api_appuser_email_8e98cebd_like
    on user_api_appuser (email varchar_pattern_ops);
```

	password	last_login	is_superuser	user_id	email	username	is_active	is_staff
1	pbkdf2_sha256 <null>		false	3	test1@gmail.com	superpuper1	• true	false
2	pbkdf2_sha256 <null>		false	4	test41@gmail.com	test1	• true	false
3	pbkdf2_sha256 2024-05-25 17:18:27.005416 +00:00		false	9	testuser@gmail.com	testuser@gmail.com	• true	false
4	pbkdf2_sha256 2024-05-26 13:34:37.804777 +00:00		false	7	new@gmail.com	new_user_new	• true	false
5	pbkdf2_sha256 2024-05-26 13:39:41.352693 +00:00		false	10	may@gmail.com	maymaymay	• true	false
6	pbkdf2_sha256 2024-05-18 20:54:14.222000 +00:00	• true		5	testUser@gmail.com	test_user	• true	false
7	pbkdf2_sha256 2024-05-22 09:56:28.395809 +00:00		false	1	test@gmail.com	superpuper	• true	false
8	pbkdf2_sha256 2024-05-25 11:08:47.332418 +00:00	• true		6	superuser@gmail.com	superuser1	• true	• true
9	pbkdf2_sha256 2024-05-25 17:06:35.364679 +00:00		false	8	mail@gmail.com	new_user	• true	false

- movie_categories

```
create table movie_categories
(
    moviecategoryid integer default
nextval('moviecategories_moviecategoryid_seq')::regclass) not null
        constraint moviecategories_pk
        primary key,
    categoryname    varchar(40)
not null
);

alter table movie_categories
owner to postgres;
```

	moviecategoryid	categoryname
1	1	Action
2	2	Comedy
3	3	Drama
4	4	Thriller
5	5	Horror
6	6	Science Fiction
7	7	Biography
8	8	Romance
9	9	Animated

- movies

```
create table movies
(
    movieid          serial
        constraint movies_pk
        primary key,
    moviecategoryid integer          not null
        constraint product_category_product
        references movie_categories,
    title            varchar(40)      not null,
    startdate        date            not null,
    enddate          date            not null,
    duration         integer          not null,
    description      varchar(255)     not null,
    image            bytea           not null,
    director         varchar(40)      not null,
    minage           integer          not null,
    production       varchar(40)      not null,
```

```

originallanguage varchar(40)      not null,
rank              double precision not null
);

alter table movies
owner to postgres;

```

	movieid	moviecategoryid	title	startdate	enddate	duration	description
1	8	9	Garfield	2024-05-31	2024-06-15	101	Garfield is the most famous cat in the world.
2	9	9	Inside Out 2	2024-06-12	2024-06-28	108	Produced by Disney and Pixar, the movie is a sequel to the original Inside Out.
3	10	9	Despicable Me 4	2024-07-05	2024-07-25	116	Gru, Lucy, Margo, Edith, and Agnes are back in this fourth installment of the Despicable Me franchise.
4	1	2	IF	2024-05-17	2024-06-01	107	From writer and director John Krasinski, this movie explores the concept of artificial intelligence.
5	2	8	Challengers	2024-04-26	2024-05-10	131	From visionary filmmaker Luca Guadagnino, this movie is set in the 1980s and follows a group of oil workers.
6	4	9	Kung Fu Panda 4	2024-03-08	2024-03-22	94	This spring, for the first time in the franchise, Po's son, Meng, takes center stage.
7	5	7	Back to Black	2024-05-17	2024-05-31	123	A behind-the-scenes glimpse into the making of the iconic 2002 film.
8	6	1	Kingdom of the Planet of the Apes	2024-05-10	2024-05-24	145	Director Wes Ball breathes new life into the Planet of the Apes franchise.
9	15	3	Dancing Queen	2024-06-07	2024-06-20	92	A 12-year-old girl who falls in love with a boy from a neighboring town.
10	11	1	Furiosa: A Mad Max Saga	2024-05-24	2024-06-06	148	The origin story of renegade warrior Furiosa.
11	3	5	Tarot	2024-05-03	2024-05-30	91	When a group of friends recklessly decide to play with fire.
12	12	3	Monster	2024-05-17	2024-06-12	126	Academy Award-nominated and Palme d'Or-winning film.
13	14	1	The Fall Guy	2024-05-03	2024-05-30	126	The film's hero is a stuntman, and he's got a secret.

- movie_halls

Może się wydawać, że wystarczyło by samo ID, ale bezpieczniej mieć osobną kolumnę, bo w przypadku usunięcia kolejny ID będzie coraz większy

```

create table movie_halls
(
    moviehallid serial
        primary key,
    hall_number integer not null
        constraint movie_halls_hall_number_unique
        unique
);

alter table movie_halls
owner to postgres;

```

	moviehallid	hall_number
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6

- movie_screening

Tabela przedstawia konkretny seans filmu

```

create table movie_screening
(
    moviescreeningid serial
        constraint movie_screening_pk
        primary key,
    movieid      integer      not null
        constraint session_movies
        references movies,
    date          date         not null,
    starttime    time         not null,
    endtime      time         not null,
    pricestandard numeric(12, 2) not null,
    pricepremium numeric(12, 2) not null,
    threedimensional boolean      not null,
    language      varchar(40)   not null,
    hallnumber    integer      not null
        constraint hallnumber
        references movie_halls (hall_number)
);
alter table movie_screening
    owner to postgres;

```

	moviescreeningid	movieid	date	starttime	endtime	pricestandard	pricepremium	threedimensional	language
1		1	2024-05-22	10:00:00	11:47:00	12.00	15.00	• true	English
2		2	2024-05-23	10:00:00	11:47:00	12.00	15.00	• true	English
3		3	2024-05-24	10:00:00	11:47:00	12.00	15.00	• true	English
4		4	2024-05-25	10:00:00	11:47:00	12.00	15.00	• true	English
5		5	2024-05-26	10:00:00	11:47:00	12.00	15.00	• true	English
6		6	2024-05-27	10:00:00	11:47:00	12.00	15.00	• true	English
7		7	2024-05-28	10:00:00	11:47:00	12.00	15.00	• true	English
8		8	2024-05-22	12:15:00	14:02:00	12.00	15.00	false	English
9		9	2024-05-23	12:15:00	14:02:00	12.00	15.00	false	English
10		10	2024-05-24	12:15:00	14:02:00	12.00	15.00	false	English
11		11	2024-05-25	12:15:00	14:02:00	12.00	15.00	false	English
12		12	2024-05-26	12:15:00	14:02:00	12.00	15.00	false	English
13		13	2024-05-27	12:15:00	14:02:00	12.00	15.00	false	English

- hall_seats

```

create table hall_seats
(
    seatid      serial
        constraint hall_seats_pk
        primary key,
    seatnumber  integer not null,
    moviehallnumber integer not null
        constraint hallseats_moviehalls
        references movie_halls (hall_number)
);
alter table hall_seats
    owner to postgres;

```

	seatid	seatnumber	moviehallnumber
1	1	1	1
2	2	2	1
3	3	3	1
4	4	4	1
5	5	5	1
6	6	6	1
7	7	7	1
8	8	8	1
9	9	9	1
10	10	10	1

- tickets

```

create table tickets
(
    ticketid          integer default nextval('tickets_tickedid_seq'::regclass) not
null
        constraint tickets_pk
            primary key,
    customerid        integer
null
        constraint client_purchase
            references user_api_appuser,
    orderedondate     date
null,
    orderedontime     time
null,
    status            char(10)
null,
    moviescreeningid integer
null
        constraint tickets_moviescreening
            references movie_screening,
    seatnumber        integer
null
        constraint tickets_hallseats
            references hall_seats
);

alter table tickets
    owner to postgres;

```

	ticketid	customerid	orderedondate	orderedontime	status	moviescreeningid	seatnumber
1	7	6	2024-05-22	13:00:00	Confirmed	1	42
2	13	6	2024-05-22	13:00:00	Confirmed	1	37
3	17	6	2024-05-22	13:00:00	Confirmed	15	40
4	23	6	2024-05-22	13:00:00	Confirmed	15	60
5	1	6	2024-05-22	13:00:00	Confirmed	1	15
6	19	6	2024-05-22	13:00:00	Canceled	22	81
7	10	6	2024-05-22	13:00:00	New	1	80
8	14	6	2024-05-22	13:00:00	Confirmed	1	6
9	16	6	2024-05-22	13:00:00	Canceled	15	6
10	20	6	2024-05-22	13:00:00	Confirmed	15	79

- trigger validate_reservation_time

```
create trigger validate_reservation_time
  before insert
  on tickets
  for each row
  execute procedure check_reservation_period();
```

4. Widoki

- Wyświetlenie danych o wszystkich zajętych miejscach na określony seans

```
create view occupied_seats(id, seatnumber, hallnumber, moviescreeningid) as
SELECT row_number() OVER (ORDER BY tickets.moviescreeningid, tickets.seatnumber)
AS id,
      tickets.seatnumber,
      movie_screening.hallnumber,
      tickets.moviescreeningid
FROM tickets
      JOIN movie_screening ON tickets.moviescreeningid =
movie_screening.moviescreeningid;

alter table occupied_seats
  owner to postgres;
```

	id	seatnumber	hallnumber	moviescreeningid
1	1	6	4	1
2	2	15	4	1
3	3	37	4	1
4	4	42	4	1
5	5	57	4	1
6	6	80	4	1
7	7	6	2	15
8	8	40	2	15
9	9	57	2	15
10	10	60	2	15

- Średnia cena biletu (standardowa i premium) dla danej kategorii wraz z liczbą seansów w okresie +- 6 miesięcy

```

create view average_ticket_prices_by_category
    (categoryname, moviescreenings_amount, average_pricestandard,
average_pricepremium) as
SELECT mc.categoryname,
    count(ms.moviescreeningid)
AS moviescreenings_amount,
    round(COALESCE(sum(ms.pricestandard) / count(ms.movieid)::numeric,
0::numeric), 2) AS average_pricestandard,
    round(COALESCE(sum(ms.pricepremium) / count(ms.movieid)::numeric,
0::numeric), 2) AS average_pricepremium
FROM movie_categories mc
    LEFT JOIN movies m ON mc.moviecategoryid = m.moviecategoryid
    LEFT JOIN movie_screening ms ON m.movieid = ms.movieid AND ms.date >=
(CURRENT_DATE - '6 mons'::interval)
GROUP BY mc.categoryname;

alter table average_ticket_prices_by_category
    owner to postgres;

```

	categoryname	moviescreenings_amount	average_pricestandard	average_pricepremium
1	Science Fiction	0	0	0
2	Romance	0	0	0
3	Drama	21	15	18
4	Horror	28	15	18
5	Action	93	16.2	19.18
6	Biography	28	15	18
7	Thriller	14	16.5	19.5
8	Comedy	28	12	15
9	Animated	0	0	0

- Zysk dla każdego filmu wraz z datą zakupu biletów

```

create view movies_revenue (title, categoryname, startdate, enddate,
orderedondate, tickets_amount, revenue) as
SELECT m.title,
    mc.categoryname,
    m.startdate,
    m.enddate,
    t.orderedondate,
    COALESCE(count(t.ticketid), 0::bigint) AS tickets_amount,
    COALESCE(sum(
        CASE
            WHEN t.ticketid IS NOT NULL AND
is_premium_place(t.seatnumber) THEN ms.pricepremium
            WHEN t.ticketid IS NOT NULL THEN ms.pricestandard
            ELSE 0::numeric
        END), 0::numeric) AS revenue
FROM movies m
    JOIN movie_categories mc ON m.moviecategoryid = mc.moviecategoryid

```

```

    LEFT JOIN movie_screening ms ON m.movieid = ms.movieid
    LEFT JOIN tickets t ON ms.moviescreeningid = t.moviescreeningid
GROUP BY m.title, mc.categoryname, m.startdate, m.enddate, t.orderedondate;

alter table movies_revenue
owner to postgres;

```

	title	categoryname	startdate	enddate	orderedondate	tickets_amount	revenue
1	Amelia's children	Horror	2024-07-12	2024-08-01	<null>	0	0
2	Back to Black	Biography	2024-05-17	2024-05-31	2024-05-20	1	18
3	Back to Black	Biography	2024-05-17	2024-05-31	2024-05-21	1	18
4	Back to Black	Biography	2024-05-17	2024-05-31	<null>	0	0
5	Bad Boys Ride or Die	Action	2024-06-07	2024-06-25	<null>	0	0
6	Challengers	Romance	2024-04-26	2024-05-10	<null>	0	0
7	Dancing Queen	Drama	2024-06-07	2024-06-20	<null>	0	0
8	Despicable Me 4	Animated	2024-07-05	2024-07-25	<null>	0	0
9	Force of Nature: The Dry 2	Thriller	2024-05-17	2024-05-23	<null>	0	0
10	Furiosa: A Mad Max Saga	Action	2024-05-24	2024-06-06	<null>	0	0
11	Garfield	Animated	2024-05-31	2024-06-15	<null>	0	0
12	Hit Man	Action	2024-05-17	2024-06-01	<null>	0	0
13	IF	Comedy	2024-05-17	2024-06-01	2024-05-22	12	150

5. Procedury

Testowanie poprawności działania procedur będzie pokazane w sekcji niżej, na przykładach z frontenu strony

- Rezerwacja miejsca na określony seans

```

create procedure reserve_movie_screening_seat(IN p_customer_id integer, IN
p_seat_number integer, IN p_movie_screening_id integer, IN p_curr_date date, IN
p_curr_time time without time zone)
language plpgsql
as
$$
BEGIN
    IF exists(select *
              from occupied_seats as oc
              where oc.MovieScreeningID = p_movie_screening_id and oc.seatnumber =
p_seat_number) THEN
        RAISE EXCEPTION 'The place has been already occupied';
    END IF;
    INSERT INTO tickets (customerid, moviescreeningid, seatnumber, orderedondate,
orderedontime, status)
    VALUES (p_customer_id, p_movie_screening_id, p_seat_number, p_curr_date,
p_curr_time, 'New');
END;
$$;

alter procedure reserve_movie_screening_seat(integer, integer, integer, date,
time) owner to postgres;

```

- Zakup miejsca na określony seans

```
create procedure buy_movie_screening_seat(IN p_customer_id integer, IN
p_seat_number integer, IN p_movie_screening_id integer, IN p_curr_date date, IN
p_curr_time time without time zone)
    language plpgsql
as
$$
BEGIN
    IF exists(select *
        from occupied_seats as oc
        where oc.MovieScreeningID = p_movie_screening_id and oc.seatnumber =
p_seat_number) THEN
        RAISE EXCEPTION 'The place has been already occupied';
    END IF;
    INSERT INTO tickets (customerid, moviescreeningid, seatnumber, orderedondate,
orderedontime, status)
        VALUES (p_customer_id, p_movie_screening_id, p_seat_number, p_curr_date,
p_curr_time, 'Confirmed');
END;
$$;

alter procedure buy_movie_screening_seat(integer, integer, integer, date, time)
owner to postgres;

create procedure update_ticket_status(IN p_ticket_id integer, IN p_new_status
character)
    language plpgsql
as
$$
BEGIN
    -- Checking the current status of the ticket
    DECLARE
        v_current_status CHAR(10);
    BEGIN
        SELECT status INTO v_current_status
        FROM tickets
        WHERE ticketid = p_ticket_id;

        -- If the current status of the ticket is "Confirmed", do not change it
        IF v_current_status = 'Confirmed' THEN
            RAISE NOTICE 'The ticket status is confirmed and cannot be changed.';
            RETURN;
        END IF;

        -- If the current status of the ticket is "New", allow changing it to
        -- "Canceled" or "Confirmed"
        IF v_current_status = 'New' THEN
            IF p_new_status = 'Canceled' OR p_new_status = 'Confirmed' THEN
                UPDATE tickets
                SET status = p_new_status
                WHERE ticketid = p_ticket_id;
                RAISE NOTICE 'Ticket status successfully updated to %.',
p_new_status;
            ELSE

```

```
        RAISE NOTICE 'The ticket status can only be changed to "Canceled"  
or "Confirmed".';  
    END IF;  
END IF;  
END;  
END;  
$$;  
  
alter procedure update_ticket_status(integer, char) owner to postgres;
```

- Dodanie nowej kategorii (admin)

```
create procedure add_movie_category(IN p_categoryname character varying)  
language plpgsql  
as  
$$  
BEGIN  
    -- Insert new category into movie_categories table  
    INSERT INTO movie_categories (categoryname)  
    VALUES (p_categoryname);  
END;  
$$;  
  
alter procedure add_movie_category(varchar) owner to postgres;  
  
create procedure delete_movie_category(IN p_moviecategoryid integer)  
language plpgsql  
as  
$$  
BEGIN  
    -- Check if the category exists  
    IF NOT EXISTS (SELECT 1 FROM movie_categories WHERE moviecategoryid =  
p_moviecategoryid) THEN  
        RAISE EXCEPTION 'Category with id % does not exist', p_moviecategoryid;  
    END IF;  
  
    -- Check if any movie exists with this category  
    IF EXISTS (SELECT 1 FROM movies WHERE moviecategoryid = p_moviecategoryid)  
THEN  
        RAISE EXCEPTION 'Cannot delete category because there are movies  
associated with it';  
    END IF;  
  
    -- Delete the category  
    DELETE FROM movie_categories  
    WHERE moviecategoryid = p_moviecategoryid;  
END;  
$$;  
  
alter procedure delete_movie_category(integer) owner to postgres;
```

- Usunięcie wybranej kategorii (admin)

```
create procedure delete_movie_category(IN p_categoryname character varying)
language plpgsql
as
$$
DECLARE
    v_moviecategoryid integer;
BEGIN
    SELECT moviecategoryid INTO v_moviecategoryid
    FROM movie_categories
    WHERE categoryname = p_categoryname;

    IF v_moviecategoryid IS NULL THEN
        RAISE EXCEPTION 'Category with name % does not exist', p_categoryname;
    END IF;

    IF EXISTS (SELECT 1 FROM movies WHERE moviecategoryid = v_moviecategoryid)
    THEN
        RAISE EXCEPTION 'Cannot delete category because there are movies
associated with it';
    END IF;

    DELETE FROM movie_categories
    WHERE moviecategoryid = v_moviecategoryid;
END;
$$;

alter procedure delete_movie_category(varchar) owner to postgres;
```

- Usunięcie wybranego filmu (admin)

```
create procedure delete_movie_by_name(IN p_movie_title text)
language plpgsql
as
$$
BEGIN
    -- Check if movie with the given name exists
    IF NOT EXISTS (SELECT 1 FROM movies WHERE title = p_movie_title) THEN
        RAISE EXCEPTION 'Movie with name % does not exist', p_movie_title;
    END IF;

    -- Delete the movie
    DELETE FROM movies
    WHERE title = p_movie_title;
END;
$$;

alter procedure delete_movie_by_name(text) owner to postgres;
```

- Dodanie nowego filmu (admin)

```
create procedure add_movie(IN p_moviecategoryname character varying, IN p_title
character varying, IN p_startdate date, IN p_enddate date, IN p_duration integer,
IN p_description character varying, IN p_image bytea, IN p_director character
varying, IN p_minage integer, IN p_production character varying, IN
p_originallanguage character varying, IN p_rank double precision)
language plpgsql
as
$$
DECLARE
    v_category_id INTEGER;
BEGIN
    -- Get movie category id from category name
    SELECT moviecategoryid INTO v_category_id FROM movie_categories WHERE
categoryname = p_moviecategoryname;

    -- Check if category id is NULL, which means no matching category found
    IF v_category_id IS NULL THEN
        RAISE EXCEPTION 'Movie category % does not exist', p_moviecategoryname;
    END IF;

    -- Check if enddate is greater than startdate
    IF p_enddate <= p_startdate THEN
        RAISE EXCEPTION 'End date must be greater than start date';
    END IF;

    -- Check if duration is at least 30 minutes
    IF p_duration < 30 THEN
        RAISE EXCEPTION 'Duration must be at least 30 minutes';
    END IF;

    -- Check if rank is between 0 and 10
    IF p_rank < 0 OR p_rank > 10 THEN
        RAISE EXCEPTION 'Rank must be between 0 and 10';
    END IF;

    -- Check if minage is between 0 and 21
    IF p_minage < 0 OR p_minage > 21 THEN
        RAISE EXCEPTION 'Minimum age must be between 0 and 21';
    END IF;

    -- Insert the new movie into the movies table
    INSERT INTO movies (
        moviecategoryid, title, startdate, enddate, duration, description,
        image, director, minage, production, originallanguage, rank
    ) VALUES (
        v_category_id, p_title, p_startdate, p_enddate, p_duration, p_description,
        p_image, p_director, p_minage, p_production, p_originallanguage, p_rank
    );
END;
$$;
```

```
alter procedure add_movie_by_name(varchar, varchar, date, date, integer, varchar,
bytea, varchar, integer, varchar, varchar, double precision) owner to postgres;
```

- Aktualizacja danych danego filmu (admin)

```
create procedure update_movie_by_name(IN p_moviecategoryname character varying, IN
p_title character varying, IN p_startdate date, IN p_enddate date, IN p_duration
integer, IN p_description character varying, IN p_image bytea, IN p_director
character varying, IN p_minage integer, IN p_production character varying, IN
p_originallanguage character varying, IN p_rank double precision)
language plpgsql
as
$$
DECLARE
    v_movie_id INTEGER;
    v_category_id INTEGER;
BEGIN
    -- Get movie ID from title
    SELECT movieid INTO v_movie_id FROM movies WHERE title = p_title;

    -- Check if movie with the given title exists
    IF v_movie_id IS NULL THEN
        RAISE EXCEPTION 'Movie with title % does not exist', p_title;
    END IF;

    -- Get movie category id from category name
    SELECT moviecategoryid INTO v_category_id FROM movie_categories WHERE
categoryname = p_moviecategoryname;

    -- Check if category name exists
    IF v_category_id IS NULL THEN
        RAISE EXCEPTION 'Movie category % does not exist', p_moviecategoryname;
    END IF;

    -- Check if end date is greater than start date
    IF p_enddate <= p_startdate THEN
        RAISE EXCEPTION 'End date must be greater than start date';
    END IF;

    -- Check if duration is at least 30 minutes
    IF p_duration < 30 THEN
        RAISE EXCEPTION 'Duration must be at least 30 minutes';
    END IF;

    -- Check if rank is between 0 and 10
    IF p_rank < 0 OR p_rank > 10 THEN
        RAISE EXCEPTION 'Rank must be between 0 and 10';
    END IF;

    -- Check if minage is between 0 and 21
    IF p_minage < 0 OR p_minage > 21 THEN
        RAISE EXCEPTION 'Minimum age must be between 0 and 21';
    END IF;

```

```

END IF;

-- Update the movie information
UPDATE movies
SET
    moviecategoryid = v_category_id,
    title = p_title,
    startdate = p_startdate,
    enddate = p_enddate,
    duration = p_duration,
    description = p_description,
    image = p_image,
    director = p_director,
    minage = p_minage,
    production = p_production,
    originallanguage = p_originallanguage,
    rank = p_rank
WHERE
    movieid = v_movie_id;
END;
$$;

alter procedure update_movie_by_name(varchar, varchar, date, date, integer,
varchar, bytea, varchar, integer, varchar, varchar, double precision) owner to
postgres;

```

- Dodanie seansów z określonym zestawem danych na określoną liczbę dni (admin) Są dodawane dokładnie takie same dane dla podanej liczby dni

```

create procedure add_movie_screenings_weekly(IN movie_title character varying, IN
start_date date, IN start_time time without time zone, IN price_standard numeric,
IN price_premium numeric, IN is_3d boolean, IN language_val character varying, IN
hall_number integer, IN repeat_count integer)
language plpgsql
as
$$
DECLARE
    movie_id integer;
    current_date_val date := start_date;
    iteration integer := 1;
BEGIN
    -- Retrieve the movie_id based on the title
    SELECT movieid INTO movie_id
    FROM movies
    WHERE title = movie_title;

    -- Check if the movie exists
    IF movie_id IS NULL THEN
        RAISE EXCEPTION 'Movie with title % does not exist', movie_title;
    END IF;

```

```

-- Check if the repeat_count is within the valid range
IF repeat_count < 1 OR repeat_count > 14 THEN
    RAISE EXCEPTION 'Repeat count % is out of the valid range (1-14)', repeat_count;
END IF;

-- Loop to add movie screenings for the specified number of repeat days
WHILE iteration <= repeat_count LOOP
    CALL add_movie_screening(
        movie_title,
        current_date_val,
        start_time,
        price_standard,
        price_premium,
        is_3d,
        language_val,
        hall_number
    );

    -- Move to the next day
    current_date_val := current_date_val + 1;
    iteration := iteration + 1;
END LOOP;
END;
$$;

alter procedure add_movie_screenings_weekly(varchar, date, time, numeric, numeric, boolean, varchar, integer, integer) owner to postgres;

```

- Dodanie nowego seansu dla danego filmu (admin)

```

create procedure add_movie_screening(IN title_val character varying, IN date_val date, IN start_time_val time without time zone, IN price_standard_val numeric, IN price_premium_val numeric, IN is_3d boolean, IN language_val character varying, IN hall_val integer)
language plpgsql
as
$$
DECLARE
    movie_id integer;
    duration_val int;
    end_time_val time;
    hall_available boolean;
BEGIN
    -- Retrieve the movie_id and duration based on the title
    SELECT m.movieid, m.duration INTO movie_id, duration_val
    FROM movies m
    WHERE m.title = title_val
    AND date_val BETWEEN m.startdate AND m.enddate; -- Check if date_val is between startdate and enddate of the movie

    -- Check if the movie exists

```

```

IF movie_id IS NULL THEN
    RAISE EXCEPTION 'Movie with title % does not exist or is not available on
the specified date', title_val;
END IF;

-- Check if the movie hall exists
IF NOT EXISTS (SELECT 1 FROM movie_halls WHERE hall_number = hall_val) THEN
    RAISE EXCEPTION 'Movie hall with id % does not exist', hall_val;
END IF;

-- Calculate the end time of the screening
SELECT calculate_end_time(start_time_val, duration_val) INTO end_time_val;

-- Check if the movie hall is available
hall_available := is_moviehall_available(hall_val, date_val, start_time_val,
end_time_val);
IF NOT hall_available THEN
    RAISE EXCEPTION 'Movie hall with id % is not available for the specified
time and date', hall_val;
END IF;

-- Insert the new movie screening record
INSERT INTO movie_screening (movieid, date, starttime, endtime, pricestandard,
pricepromium, threedimensional, language, hallnumber)
VALUES (movie_id, date_val, start_time_val, end_time_val, price_standard_val,
price_premium_val, is_3d, language_val, hall_val);
END
$$;

alter procedure add_movie_screening(varchar, date, time, numeric, numeric,
boolean, varchar, integer) owner to postgres;

```

- Usunięcie wybranego seansu (admin)

```

create procedure delete_movie_screening(IN movie_title text, IN screening_date
date, IN screening_time time without time zone)
language plpgsql
as
$$
DECLARE
    screening_id INTEGER;
BEGIN
    -- Перевірка наявності показу фільму з заданими параметрами
    SELECT ms.moviescreeningid
    INTO screening_id
    FROM movie_screening ms
    JOIN movies m ON ms.movieid = m.movieid
    WHERE m.title = movie_title
    AND ms.date = screening_date
    AND ms.starttime = screening_time;

    -- Якщо показ не знайдено, викликаємо помилку

```

```

IF NOT FOUND THEN
    RAISE EXCEPTION 'No movie screening for the movie "%", date "%", and time
"%" exists', movie_title, screening_date, screening_time;
END IF;

-- Видалення знайденого показу
DELETE FROM movie_screening
WHERE moviescreeningid = screening_id;
END
$$;

alter procedure delete_movie_screening(text, date, time) owner to postgres;

```

6. Funkcje

- Wyświetlenie wszystkich seansów dla danego movie, które są grane po wskazywanym terminie

```

create function get_movie_sessions(p_title character varying, target_date date,
target_time time without time zone)
    returns TABLE(moviescreeningid integer, movieid integer, date date, starttime
time without time zone, pricestandard numeric, pricepremium numeric, moviehall
integer, threedimensional boolean, language character varying)
    language plpgsql
as
$$
BEGIN
    RETURN QUERY
    SELECT
        movie_screening.moviescreeningid,
        movie_screening.movieid,
        movie_screening.date,
        movie_screening.starttime,
        movie_screening.pricestandard,
        movie_screening.pricepremium,
        movie_screening.hallnumber,
        movie_screening.threedimensional,
        movie_screening.language
    FROM
        movie_screening
    INNER JOIN
        movies as m on movie_screening.MovieID = m.movieid
    WHERE
        m.title = get_movie_sessions.p_title
        AND (movie_screening.date > get_movie_sessions.target_date
        OR (movie_screening.date >= get_movie_sessions.target_date AND
movie_screening.starttime > get_movie_sessions.target_time));
END;
$$;

alter function get_movie_sessions(varchar, date, time) owner to postgres;

```

- Wyświetlenie wszystkich zajętych miejsc dla danego seansu

```

CREATE OR REPLACE FUNCTION get_occupied_seats(screening_id integer)
RETURNS TABLE(seat_number integer)
LANGUAGE plpgsql
AS
$$
BEGIN
    RETURN QUERY
    SELECT
        seatnumber AS seat_number
    FROM
        occupied_seats
    WHERE
        moviescreeningid = get_occupied_seats.screening_id;
END;
$$;

ALTER FUNCTION get_occupied_seats(integer) OWNER TO postgres;

```

- Wyświetlenie wszystkich filmów aktualnie granych w kinie

```

create function get_current_movies(p_date date)
    returns TABLE(movieid integer, moviecategoryid integer, title character
varying, startdate date, enddate date, duration integer, description character
varying, image bytea, director character varying, minage integer, production
character varying, originallanguage character varying, rank double precision)
    language plpgsql
as
$$
BEGIN
    RETURN QUERY
    SELECT
        m.movieid,
        m.moviecategoryid,
        m.title,
        m.startdate,
        m.enddate,
        m.duration,
        m.description,
        m.image,
        m.director,
        m.minage,
        m.production,
        m.originallanguage,
        m.rank
    FROM
        movies m
    WHERE
        m.startdate <= p_date and p_date <= m.enddate;

```

```
END;
$$;

alter function get_current_movies(date) owner to postgres;
```

- Wyświetlenie wszystkich filmów, które będą grane w przyszłości

```
create function get_upcoming_movies(p_date date)
  returns TABLE(movieid integer, moviecategoryid integer, title character
varying, startdate date, enddate date, duration integer, description character
varying, image bytea, director character varying, minage integer, production
character varying, originallanguage character varying, rank double precision)
  language plpgsql
as
$$
BEGIN
  RETURN QUERY
  SELECT
    m.movieid,
    m.moviecategoryid,
    m.title,
    m.startdate,
    m.enddate,
    m.duration,
    m.description,
    m.image,
    m.director,
    m.minage,
    m.production,
    m.originallanguage,
    m.rank
  FROM
    movies m
  WHERE
    p_date < m.startdate;
END;
$$;

alter function get_upcoming_movies(date) owner to postgres;
```

- Sprawdzenie, czy sala jest dostępna w określonym terminie

```
create function is_moviehall_available(hall_val integer, date_val date,
start_time_val time without time zone, end_time_val time without time zone)
returns boolean
language plpgsql
as
$$
DECLARE
```

```

is_available boolean;
BEGIN
  SELECT NOT EXISTS (
    SELECT 1 FROM movie_screening
    WHERE hallnumber = hall_val
    AND date = date_val
    AND (starttime <= end_time_val AND endtime >= start_time_val)
  ) INTO is_available;

  RETURN is_available;
END;
$$;

alter function is_moviehall_available(integer, date, time, time) owner to
postgres;

```

- Obliczanie end_time dla seansu (jest potrzebne do wstawienia danych w move_screenings)

```

create function calculate_end_time(start_time_param time without time zone,
duration_param integer) returns time without time zone
language plpgsql
as
$$
DECLARE
  end_time_result TIME;
BEGIN
  SELECT (start_time_param + INTERVAL '1 minute' * duration_param) INTO
end_time_result;
  RETURN end_time_result;
END;
$$;

alter function calculate_end_time(time, integer) owner to postgres;

```

- Wyświetlenie terminów wszystkich seansów prowadzonych w danej sali

```

create function get_screenings_by_hall(hall_number_param integer, date_param date)
returns TABLE(movieid integer, title character varying, starttime time without
time zone, endtime time without time zone)
language plpgsql
as
$$
BEGIN
  RETURN QUERY
  SELECT DISTINCT
    ms.movieid,
    m.title,
    ms.starttime,
    ms.endtime

```

```

FROM
    movie_screening ms
INNER JOIN movies m
    on ms.movieid = m.movieid
WHERE
    ms.hallnumber = hall_number_param
    AND ms.date = date_param
ORDER BY
    ms.starttime;
END;
$$;

alter function get_screenings_by_hall(integer, date) owner to postgres;

```

- Wyświetlenie wszystkich biletów dla danego użytkownika

```

create function get_tickets_for_user(user_id integer)
    returns TABLE(ticket_id integer, status character, date date, title character
varying, start_time time without time zone, duration integer, hall_number integer,
sit_number integer, price numeric, ordered_on_date date, ordered_on_time time
without time zone)
    language plpgsql
as
$$
begin
    return query
    select
        t.ticketid,
        t.status,
        ms.Date,
        m.title,
        ms.StartTime,
        m.duration,
        ms.hallnumber,
        t.seatnumber,
        CASE
            WHEN is_premium_place(t.seatnumber) THEN ms.PricePremium
            ELSE ms.PriceStandard
        END AS price,
        t.orderedondate,
        t.orderedontime
    from
        tickets t
    inner join
        movie_screening as ms on t.MovieScreeningID = ms.MovieScreeningID
    inner join
        movies as m on m.movieid = ms.movieid
    where
        t.customerid = user_id
    order by date, starttime;
end;
$$;

```

```
alter function get_tickets_for_user(integer) owner to postgres;
```

- Sprawdzenie, czy dane miejsce jest miejscem z kategorii Premium (wszystkie sale mają tyle samo miejsc, ostatni rząd symbolizuje premium miejsca)

```
create function is_premium_place(place_value integer) returns boolean
language plpgsql
as
$$
BEGIN
    IF place_value >= 1 AND place_value <= 11 THEN
        RETURN TRUE;
    ELSE
        RETURN FALSE;
    END IF;
END;
$$;

alter function is_premium_place(integer) owner to postgres;
```

- Wyświetlenie danych o filmie wraz z nazwą kategorii

```
create function get_movie_by_title(p_title character varying)
    returns TABLE(movieid integer, categoryname character varying, title character
varying, startdate date, enddate date, duration integer, description character
varying, image bytea, director character varying, minage integer, production
character varying, originallanguage character varying, rank double precision)
language plpgsql
as
$$
BEGIN
    RETURN QUERY
    SELECT
        m.movieid,
        mc.categoryname,
        m.title,
        m.startdate,
        m.enddate,
        m.duration,
        m.description,
        m.image,
        m.director,
        m.minage,
        m.production,
        m.originallanguage,
        m.rank
    FROM
        movies m
```

```

    INNER JOIN
        movie_categories as mc
    on m.moviecategoryid = mc.moviecategoryid
    WHERE p_title = m.title;
END;
$$;

alter function get_movie_by_title(varchar) owner to postgres;

```

- Wyświetlenie wszystkich dostępnych w danym dniu seansów wraz z nazwą filmu

```

create function get_movie_screenings_on_date(target_date date)
    returns TABLE(movie_title character varying, start_time time without time
zone, hall_number integer, available_seats integer)
    language plpgsql
as
$$
BEGIN
    RETURN QUERY
        SELECT m.title, ms.starttime, ms.hallnumber, ((SELECT COUNT(*) FROM hall_seats
WHERE moviehallnumber = ms.hallnumber) - COALESCE((SELECT COUNT(*) FROM tickets
WHERE moviescreeningid = ms.moviescreeningid), 0))::integer AS available_seats
        FROM movies m
        INNER JOIN movie_screening ms ON m.movieid = ms.movieid
        WHERE ms.date = target_date;
END;
$$;

alter function get_movie_screenings_on_date(date) owner to postgres;

```

7. Triggery

- Zabronienie rezerwacji na seans który jest grany za mniej niż 2 godziny (możliwy jest wyłącznie ZAKUP biletu na taki seans)

```

create function check_reservation_period() returns trigger
    language plpgsql
as
$$
DECLARE
    screening_date DATE;
    screening_start_time TIME;
BEGIN
    SELECT date, starttime
    INTO screening_date, screening_start_time
    FROM movie_screening
    WHERE MovieScreeningID = NEW.MovieScreeningID;

    IF NEW.Status = 'New' and (screening_date + screening_start_time) <=

```

```

(NEW.OrderedOnDate + NEW.OrderedOnTime + interval '2 hours') THEN
    RAISE EXCEPTION 'The movie screening must be later than 2 hours from the
reservation time';
END IF;

RETURN NEW;
END;
$$;

alter function check_reservation_period() owner to postgres;

```

```

CREATE TRIGGER validate_reservation_time
BEFORE INSERT
ON tickets
FOR EACH ROW
EXECUTE FUNCTION check_reservation_period();

```

8. Indeksy

```

-- Tabela Appuser
CREATE INDEX idx_appuser_email ON user_api_appuser(email);
CREATE INDEX idx_appuser_username ON user_api_appuser(username);

-- Tabela HallSeats
CREATE INDEX idx_hallseats_moviehallnumber ON hall_seats(moviehallnumber);

-- Tabela MovieScreening
CREATE INDEX idx_moviescreening_movieid ON movie_screening(movieid);
CREATE INDEX idx_moviescreening_date ON movie_screening(date);
CREATE INDEX idx_moviescreening_starttime ON movie_screening(starttime);

-- Tabela Movies
CREATE INDEX idx_movies_moviecategoryid ON movies(moviecategoryid);
CREATE INDEX idx_movies_title ON movies(title);
CREATE INDEX idx_movies_startdate ON movies(startdate);
CREATE INDEX idx_movies_enddate ON movies(enddate);

-- Tabela Tickets
CREATE INDEX idx_tickets_customerid ON tickets(customerid);
CREATE INDEX idx_tickets_moviescreeningid ON tickets(moviescreeningid);
CREATE INDEX idx_tickets_seatnumber ON tickets(seatnumber);

```

9. Widoki strony internetowej

Strona główna (home)

Na stronie są pokazane grane teraz i w przyszłości filmy, działa wyszukiwanie filmu

CineVerse

Home About us Login Movies ⌂

Welcome.

Enchanting stories come to life on the silver screen, offering a journey of emotions and adventures for every movie enthusiast.

Search for movie...

Now Playing

IF

Back to Black

Kingdom of the Planet of the Apes

Tarot

Monster

The Fall Guy

Hit Man

Coming Soon

Garfield

Inside Out 2

Despicable Me 4

Dancing Queen

Furiosa: A Mad Max Saga

Bad Boys Ride or Die

Turbo

Movies

Na stronie są wyświetlane wszystkie movies dostępne teraz, oraz movies, które będą dostępne w przyszłości. Sortowanie domyślnie ustawione od daty premiery filmu. Na stronie działa wyszukiwanie po nazwie filmu, filtrowanie po kategorii, sortowanie według zadanych parametrów

Screenshot of the CineVerse movie database interface. The top navigation bar includes 'Home', 'About us', 'Login', 'Movies', a user icon, and a shopping cart icon. Below the navigation is a search bar labeled 'Search movies...', a 'Select genres' dropdown, and a 'Sort by' dropdown. The main content area displays a grid of movie cards, each with a thumbnail, title, and a brief description. The movies shown are:

- The Fall Guy**: Ryan Reynolds, Emily Blunt. Only in theaters May 3.
- Tarot**: YOUR FATE IS IN THE CARDS. ONLY IN THEATERS MAY 10.
- Kingdom of the Planet of the Apes**: Jeff Bridges, Lena Waithe. ONLY IN THEATERS MAY 10.
- Hit Man**: Ben Powell, Boris Aljinovic. ONLY IN THEATERS MAY 10.
- Back to Black**: HER MUSIC. HER LIFE. HER TERMS. ONLY IN THEATERS MAY 12.
- Force of Nature: The Dry 2**: Eric Bana. EXCLUSIVE TO CINEMAS TUESDAY 9.
- IF**: A STORY YOU HAVE TO BELIEVE TO SEE. ONLY IN THEATERS MAY 17.
- Monster**: NANA Mouskouri. ONLY IN THEATERS MAY 17.
- Turbo**: A MAMMOTTY FILM. ONLY IN THEATERS MAY 17.
- Furiosa: A Mad Max Saga**: Anya Taylor-Joy, Chris Hemsworth. FURY ROAD MEETS MAD MAX. ONLY IN THEATERS MAY 24.
- Garfield**: TRZYMAJ SIE FOTELA! ONLY IN THEATERS MAY 24.
- Bad Boys Ride or Die**: Will Smith, Martin Lawrence. WILL MAŁEGÓDĘ TĘKAĆ GA OSOBAMI. ONLY IN THEATERS MAY 24.
- Dancing Queen**: ONLY IN THEATERS MAY 24.
- TEST**: ONLY IN THEATERS MAY 24.
- Inside Out 2**: ONLY IN THEATERS JUN 14.
- Despicable Me 4**: ONLY IN THEATERS JULY 3.
- Amelia's Children**: ONLY IN THEATERS JULY 11.

Movie

Strona wybranego filmu, gdzie są wyświetlane informacje dotyczące danego filmu wraz z kategorią. Jest możliwość przeglądania dostępnych w ciągu przyszłych 7 dni terminów seansów i przejście do strony danego seansu (opisana niżej)

CineVerse

Home About us Login Movies  



IF

COMEDY

From writer and director John Krasinski, **IF** is about a girl who discovers that she can see everyone's imaginary friends — and what she does with that superpower — as she embarks on a magical adventure to reconnect forgotten **IFs** with their kids. **IF** stars R

Start Date: 5/17/2024

End Date: 6/1/2024

Duration: 107 minutes

Director: John Krasinski

Minimum Age: 12

Production: Paramount Pictures

Original Language: English

Buy Ticket

BUY TICKET ON IF

Wed Thu Fri Sat Sun Mon Tue

Date: 5/23/2024 Start: 10:00:00 3D: Yes Language: English Movie Hall: 4 **Choose perfect seat**

Date: 5/23/2024 Start: 12:15:00 3D: No Language: English Movie Hall: 2 **Choose perfect seat**

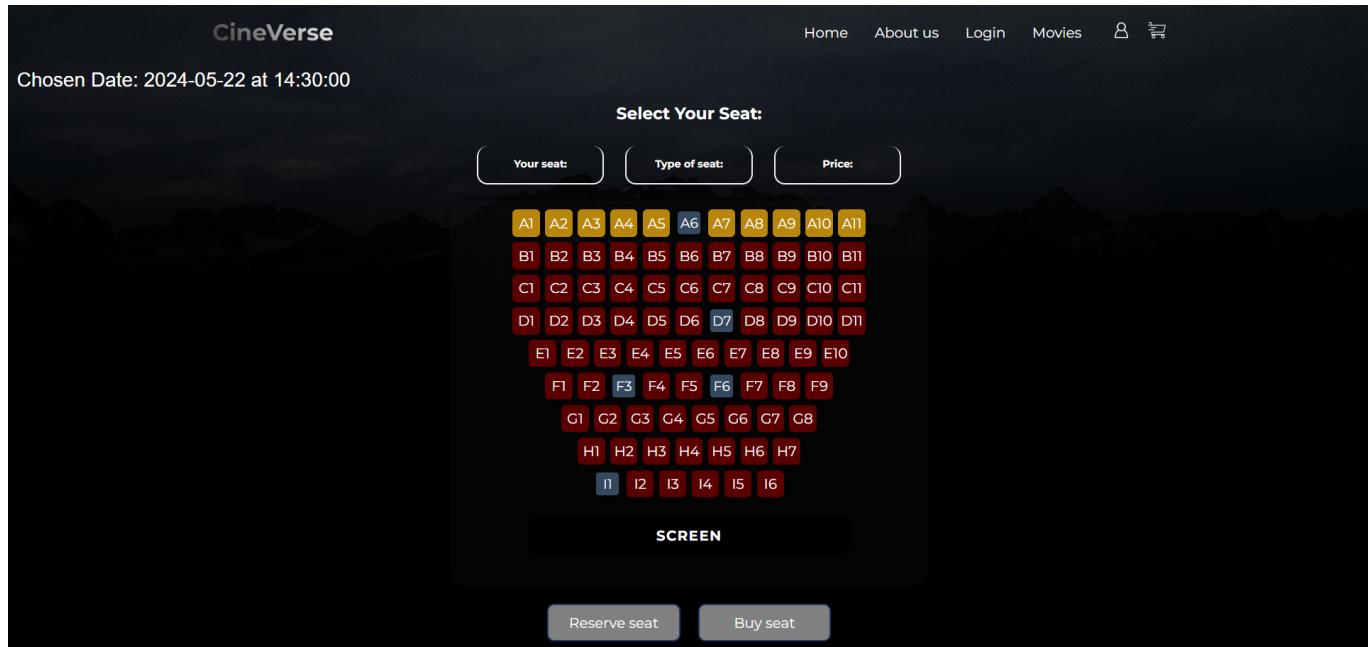
Date: 5/23/2024 Start: 14:30:00 3D: No Language: English Movie Hall: 2 **Choose perfect seat**

Date: 5/23/2024 Start: 16:45:00 3D: Yes Language: English Movie Hall: 3 **Choose perfect seat**

Showtime

Strona wybranego seansu filmu, gdzie są wyświetlane informacje dotyczące danego seansu. Klient ma możliwość podglądu zajętych miejsc na dany seans (są zaznaczone na szaro), oraz wyboru miejsca wśród Standard (12-84) albo Premium (1-11) miejsc, które też są zaznaczone różnymi kolorami i mają różne ceny

Początkowy wygląd strony:



CineVerse

Home About us Login Movies

Chosen Date: 2024-05-22 at 14:30:00

Select Your Seat:

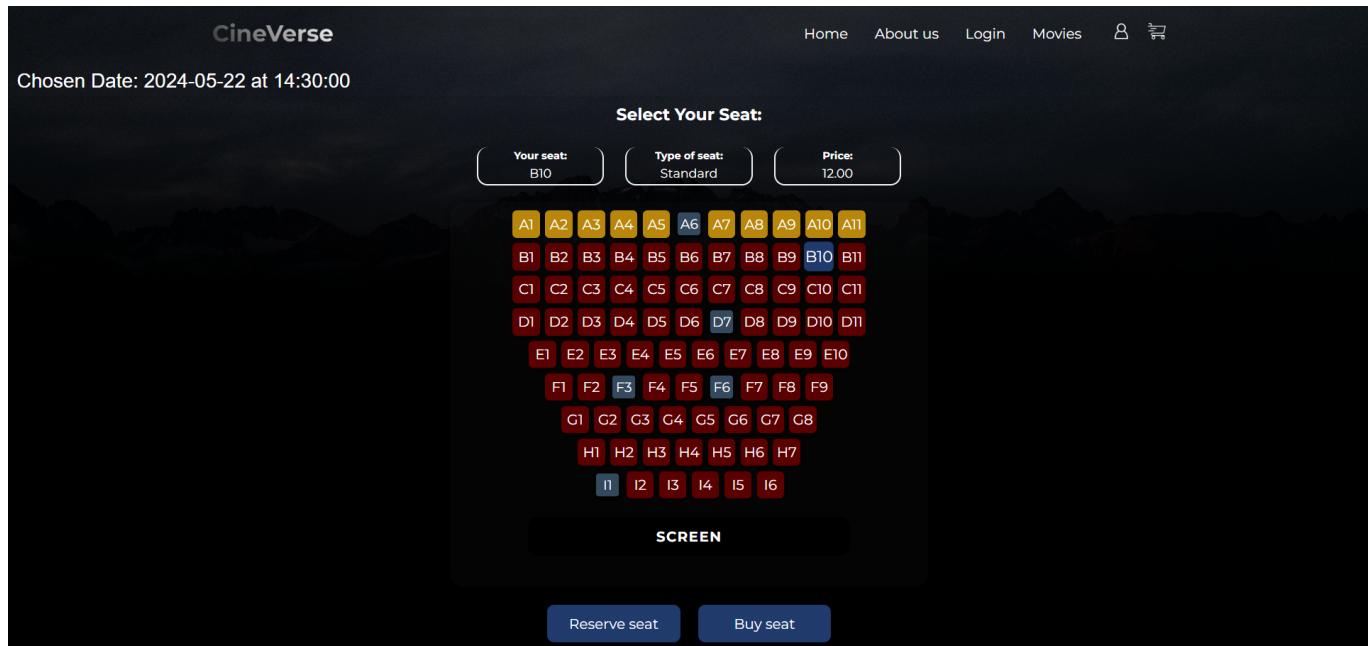
Your seat: A1 Type of seat: Standard Price: 12.00

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	
F1	F2	F3	F4	F5	F6	F7	F8	F9		
G1	G2	G3	G4	G5	G6	G7	G8			
H1	H2	H3	H4	H5	H6	H7				
I1	I2	I3	I4	I5	I6					

SCREEN

Reserve seat Buy seat

Wygląd strony po kliknięciu na miejsce:



CineVerse

Home About us Login Movies

Chosen Date: 2024-05-22 at 14:30:00

Select Your Seat:

Your seat: B10 Type of seat: Standard Price: 12.00

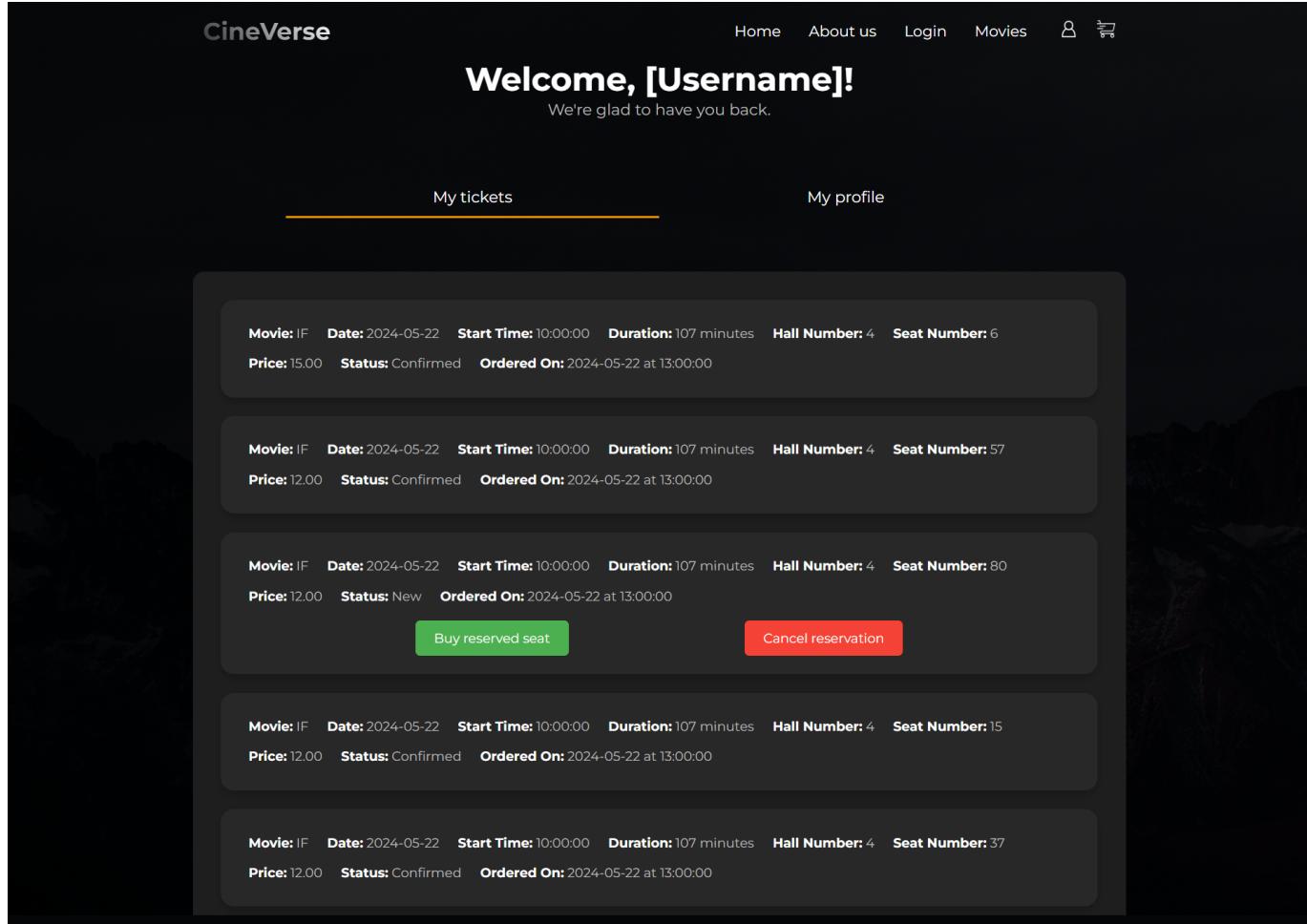
A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	
F1	F2	F3	F4	F5	F6	F7	F8	F9		
G1	G2	G3	G4	G5	G6	G7	G8			
H1	H2	H3	H4	H5	H6	H7				
I1	I2	I3	I4	I5	I6					

SCREEN

Reserve seat Buy seat

UserProfile

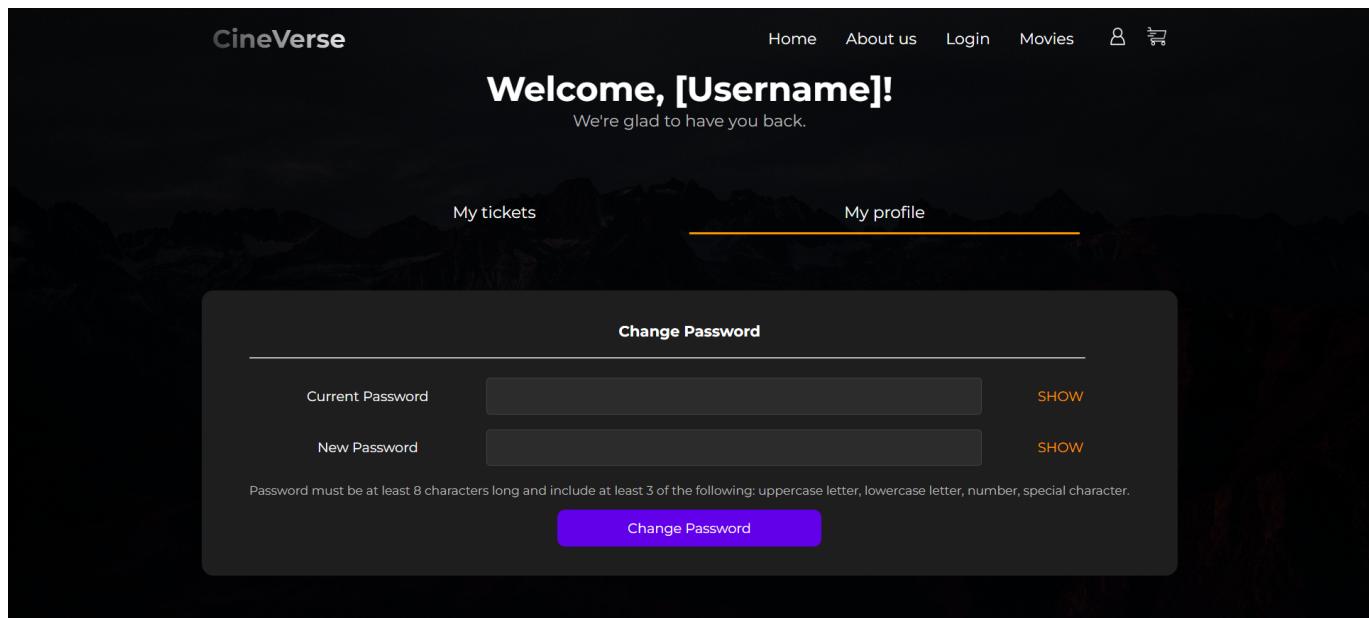
Strona profilu użytkownika, gdzie są 2 przełączniki pomiędzy ticketami i ustawieniami. W zakładce "My tickets" są wyświetlane wszystkie tickety klienta wraz z informacją o seansie. W contenerze biletów, mających status "New", jest możliwość zakupu danego biletu lub jego anulowania. W zakładce "Settings" istnieje możliwość zmiany hasła użytkownika (na razie nie zrealizowana technicznie)



The screenshot shows the 'My tickets' section of the CineVerse application. It displays a list of movie tickets with the following details:

- Movie: IF Date: 2024-05-22 Start Time: 10:00:00 Duration: 107 minutes Hall Number: 4 Seat Number: 6**
Price: 15.00 Status: Confirmed Ordered On: 2024-05-22 at 13:00:00
- Movie: IF Date: 2024-05-22 Start Time: 10:00:00 Duration: 107 minutes Hall Number: 4 Seat Number: 57**
Price: 12.00 Status: Confirmed Ordered On: 2024-05-22 at 13:00:00
- Movie: IF Date: 2024-05-22 Start Time: 10:00:00 Duration: 107 minutes Hall Number: 4 Seat Number: 80**
Price: 12.00 Status: New Ordered On: 2024-05-22 at 13:00:00
- Movie: IF Date: 2024-05-22 Start Time: 10:00:00 Duration: 107 minutes Hall Number: 4 Seat Number: 15**
Price: 12.00 Status: Confirmed Ordered On: 2024-05-22 at 13:00:00
- Movie: IF Date: 2024-05-22 Start Time: 10:00:00 Duration: 107 minutes Hall Number: 4 Seat Number: 37**
Price: 12.00 Status: Confirmed Ordered On: 2024-05-22 at 13:00:00

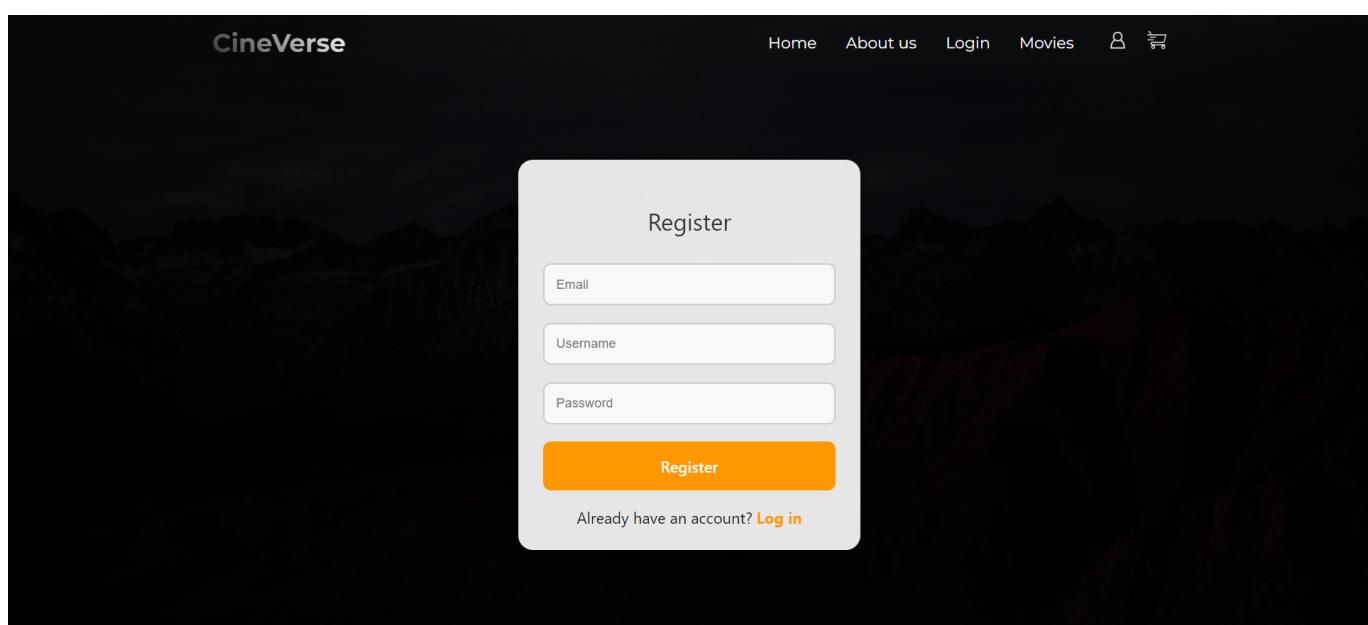
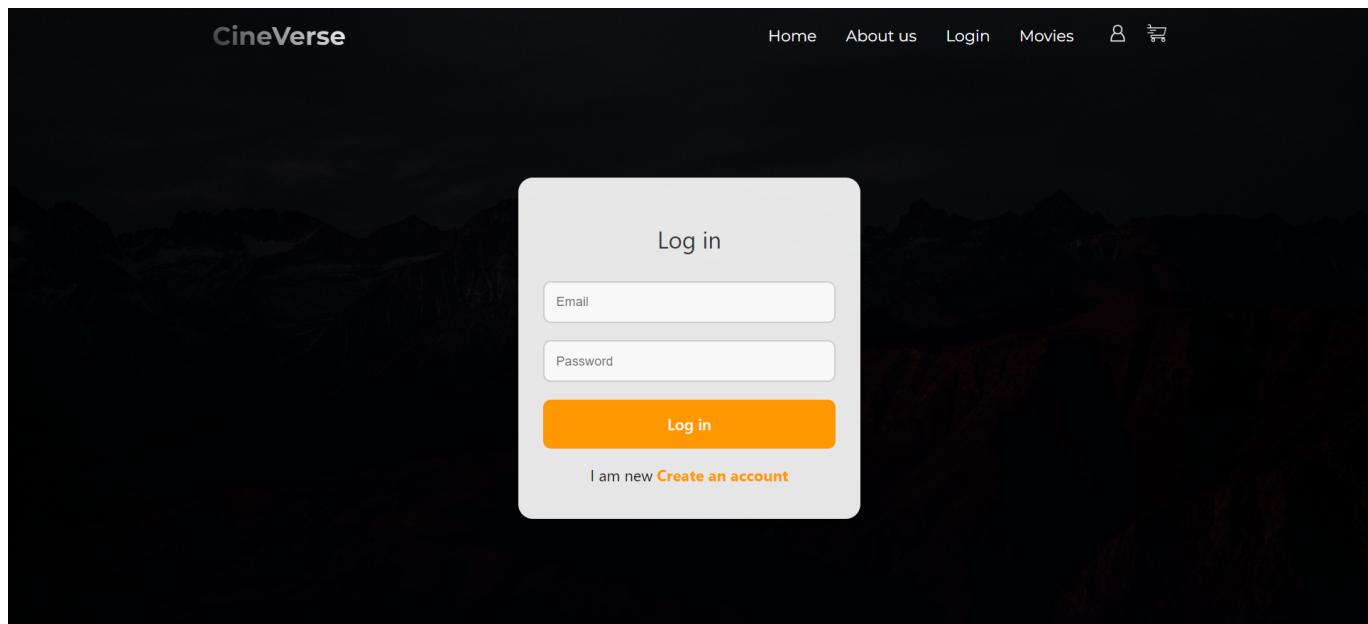
For the third ticket (Seat 80), there are two buttons: 'Buy reserved seat' (green) and 'Cancel reservation' (red).



The screenshot shows the 'Change Password' settings section of the CineVerse application. It includes fields for 'Current Password' and 'New Password', each with a 'SHOW' link to reveal the entered text. A note at the bottom states: "Password must be at least 8 characters long and include at least 3 of the following: uppercase letter, lowercase letter, number, special character." A 'Change Password' button is located at the bottom of the form.

Login & Register

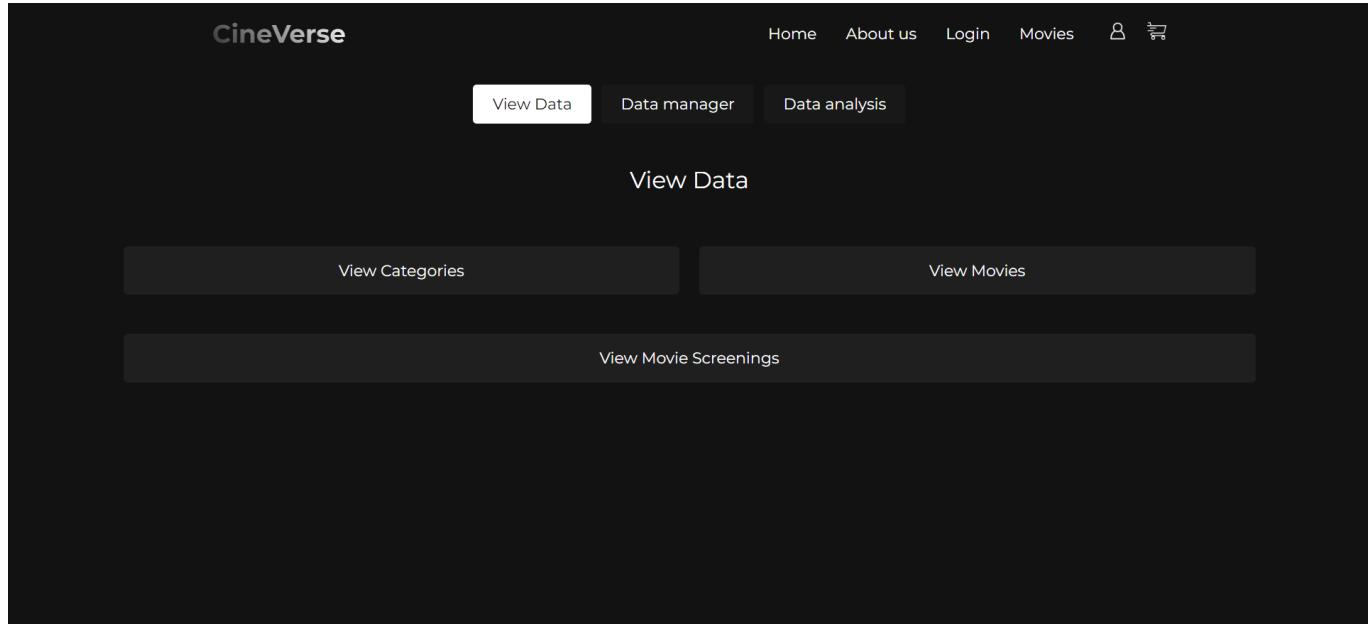
Na razie logowanie i rejestracja działają na poziomie bazy danych (pojawiają się odpowiednie dane w tabelu userów), natomiast w tym momencie nie działa ustawienie sesji użytkownika



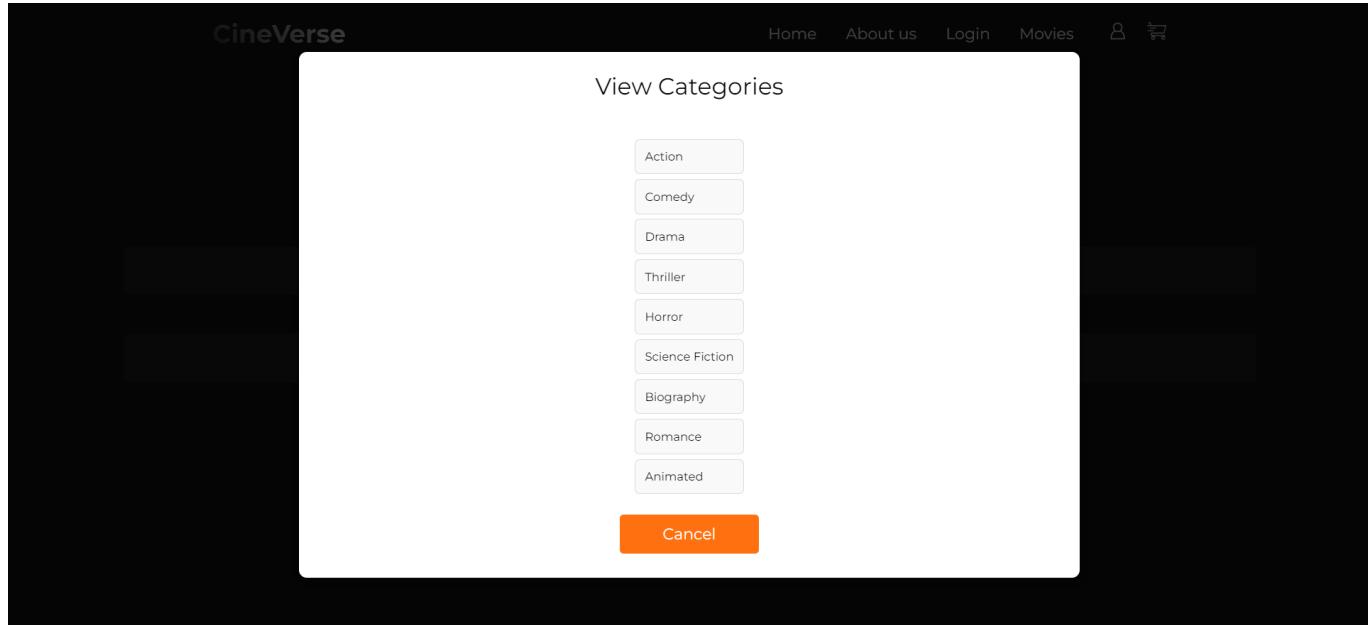
Admin

Strona posiada 3 przełączniki: "View Data" - podgląd danych z tabel, "Data manager" - zarządzanie danymi tabel (dodawanie rekordów, ich modyfikacja oraz usunięcie), "Data analysis" - podgląd statystycznych danych bez możliwości modyfikacji (wywołania widoku lub funkcji, niektóre mają parametry, a niektóre nie).

- View Data



Categories:



Movies:

CineVerse

Home About us Login Movies  

View Movies

Hit Man
Back to Black
Force of Nature: The Dry 2
IF
Monster
Turbo
Furiosa: A Mad Max Saga
TEST2
Garfield

[Cancel](#)

CineVerse

Home About us Login Movies  

Despicable Me 4



Category: Animated
Start date: 2024-07-05
End date: 2024-07-25
Duration: 116
Description: Gru, Lucy, Margo, Edith, and Agnes welcome a new member to the family, Gru Jr., who is intent on tormenting his dad. Gru faces a new nemesis in Maxime Le Mal and his girlfriend Valentina, and the family is forced to go on the run.
Minage: 10
Director: Emma Smith
Production: USA 2024
Original language: English
Rank: 9.1

[Back](#)

Movie Screenings:

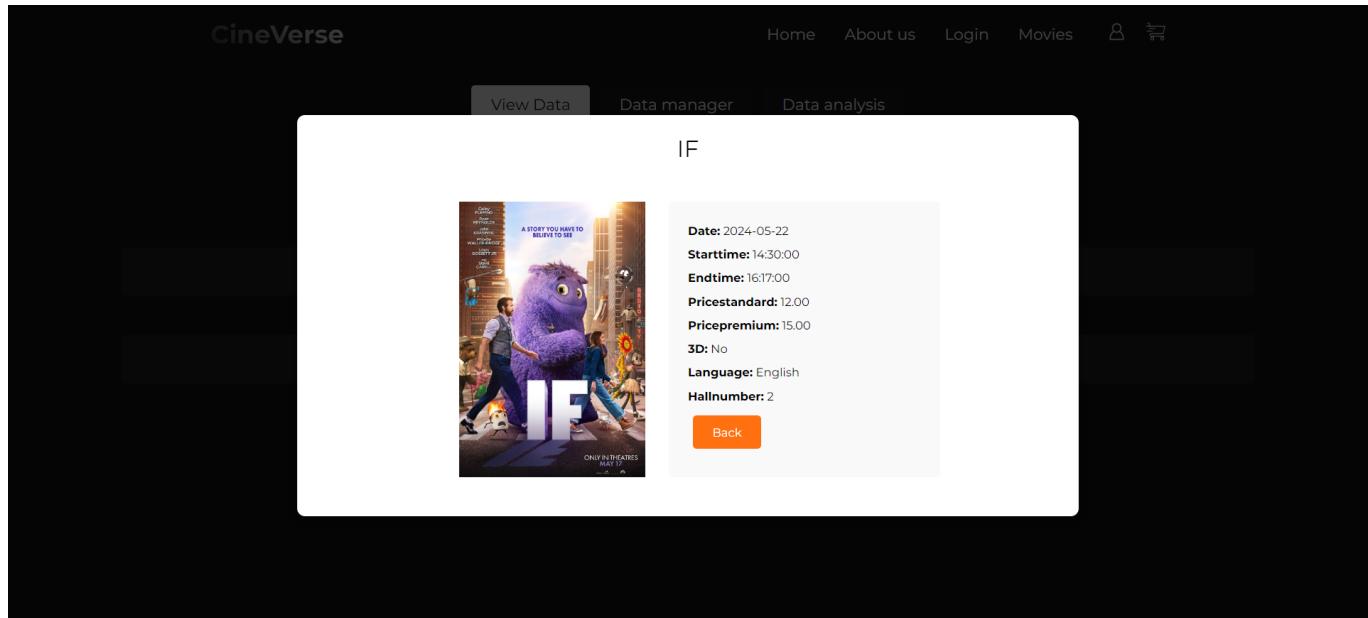
CineVerse

Home About us Login Movies  

View Movie Screenings

IF - 2024-05-22 10:00:00
Hit Man - 2024-05-22 12:00:00
IF - 2024-05-22 12:15:00
Kingdom of the Planet of the Apes - 2024-05-22 12:15:00
Back to Black - 2024-05-22 12:20:00
Kingdom of the Planet of the Apes - 2024-05-22 14:30:00
IF - 2024-05-22 14:30:00
Back to Black - 2024-05-22 14:45:00

[Cancel](#)



- Data manager

The screenshot shows the 'Data manager' section of the CineVerse application. At the top, there is a navigation bar with links for Home, About us, Login, Movies, a user icon, and a shopping cart icon. Below the navigation bar, there are three tabs: View Data, Data manager, and Data analysis. The 'Data manager' tab is currently selected. The main content area is divided into several sections:

- MovieCategories:** Contains buttons for 'Add New Category' and 'Delete Category'.
- Movies:** Contains buttons for 'Add New Movie', 'Update Movie', and 'Delete Movie'.
- MovieScreening:** Contains buttons for 'Add Week-Templated Movie Screenings' and 'Delete Movie Screening'.

Przykładowo - Add Week-Templated Movie Screenings:

MovieCategories

Add Week-Templated Movie Screenings

Movie title: Price standard:

Date: Price premium:

Start time: Language:

Is 3D?: Movie hall:

How much days since given date?

Confirm **Cancel**

Add Week-Templated Movie Screenings Delete Movie Screening

- Data analysis

CineVerse

Home About us Login Movies User icon

Data analysis

MovieScreening

Show Movie Screenings by Hall Show revenue for Movie on OrderDate

Show average prices per category over past/upcoming 6 months Show today Movie Screenings

Show Movie Screenings by Hall:

CineVerse

Show Movie Screenings by Hall

Hall number: Date:

Confirm **Cancel**

Title: Hit Man
StartTime: 12:00:00
EndTime: 13:55:00

Title: Tarot
StartTime: 15:00:00
EndTime: 16:31:00

Title: IF
StartTime: 16:45:00
EndTime: 18:34:00

Title: Kingdom of the Planet of the Apes
StartTime: 18:45:00
EndTime: 21:00:00

Title: Monster
StartTime: 21:30:00
EndTime: 23:36:00

Show revenue for Movie on OrderDate:

CineVerse

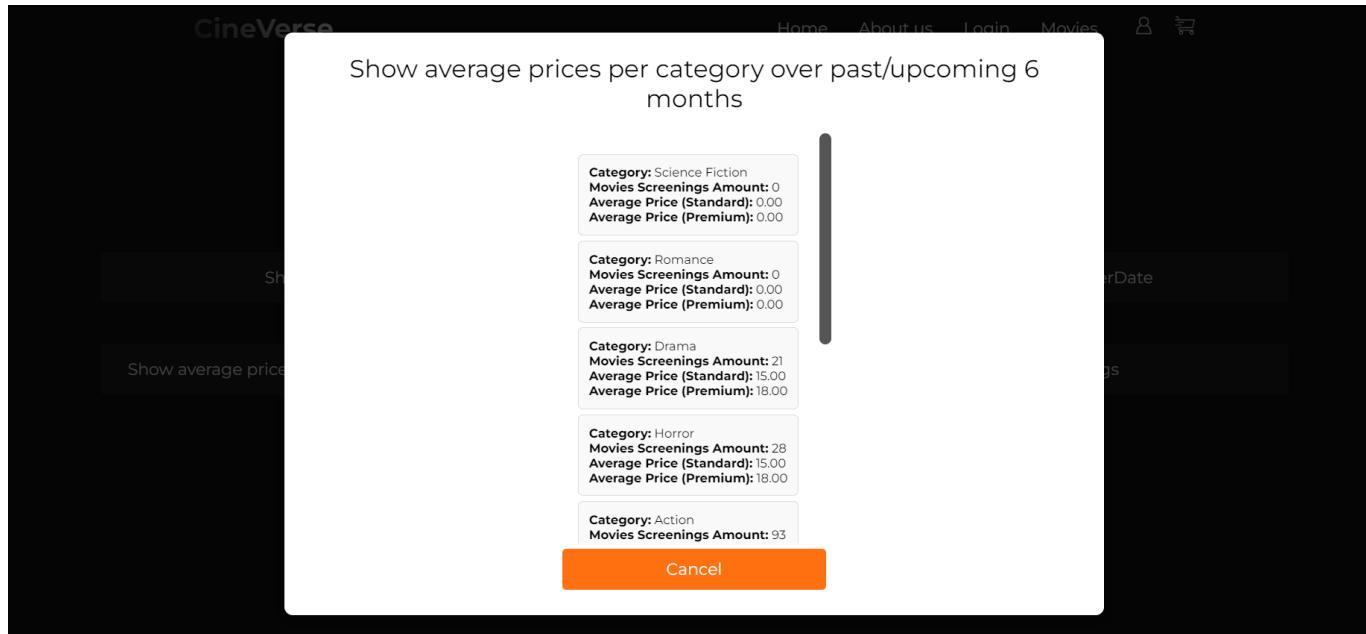
Show revenue for Movie on OrderDate

Movie title: Ordered on date:

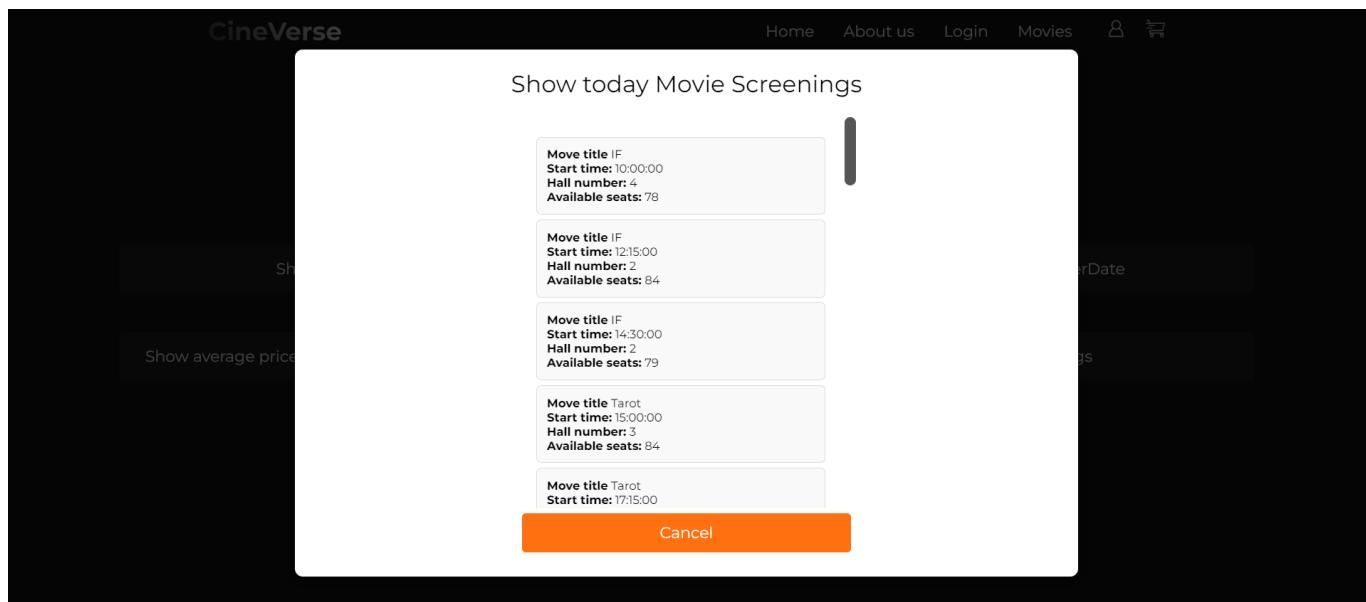
Confirm **Cancel**

Categoryname: Comedy
Startdate: 2024-05-17
Enddate: 2024-06-01
Ordered on date: 2024-05-22
Tickets bought: 12
Revenue: 150.00

Show average prices per category over past/upcoming 6 months:



Show today Movie Screenings:



10. Testowanie poprawności działania procedur na stronie

(Test został wykonany na wcześniejszej wersji frontendu jednak działanie bazy danych nie zmieniło się)

Użytkownik

Najpierw przetestujemy rezerwację i zakup biletu, zróbmy to dla przykładu na jednym seansie, przy czym weźmiemy różne miejsca, żeby sprawdzić, czy cena będzie się różniła.

- Rezerwacja biletu

Tabela tickets przed wywołaniem procedury:

	ticketid	customerid	orderedondate	orderedontime	status	moviescreeningid	seatnumber
1		25	6 2024-05-21	16:30:00	Confirmed	64	10
2		24	6 2024-05-20	14:30:00	Confirmed	63	10
3		23	6 2024-05-22	13:00:00	Confirmed	15	60
4		22	6 2024-05-22	13:00:00	Confirmed	15	57
5		20	6 2024-05-22	13:00:00	Confirmed	15	79
6		19	6 2024-05-22	13:00:00	Canceled	22	81
7		17	6 2024-05-22	13:00:00	Confirmed	15	40
8		16	6 2024-05-22	13:00:00	Canceled	15	6
9		14	6 2024-05-22	13:00:00	Confirmed	1	6
10		13	6 2024-05-22	13:00:00	Confirmed	1	37

Wybrane przez nas miejsce F4, którego cena jest Standard:

Standard price: 12.00
Premium price: 15.00

Select Your Seat:
Your seat: F4

SCREEN

Reserve seat Buy seat

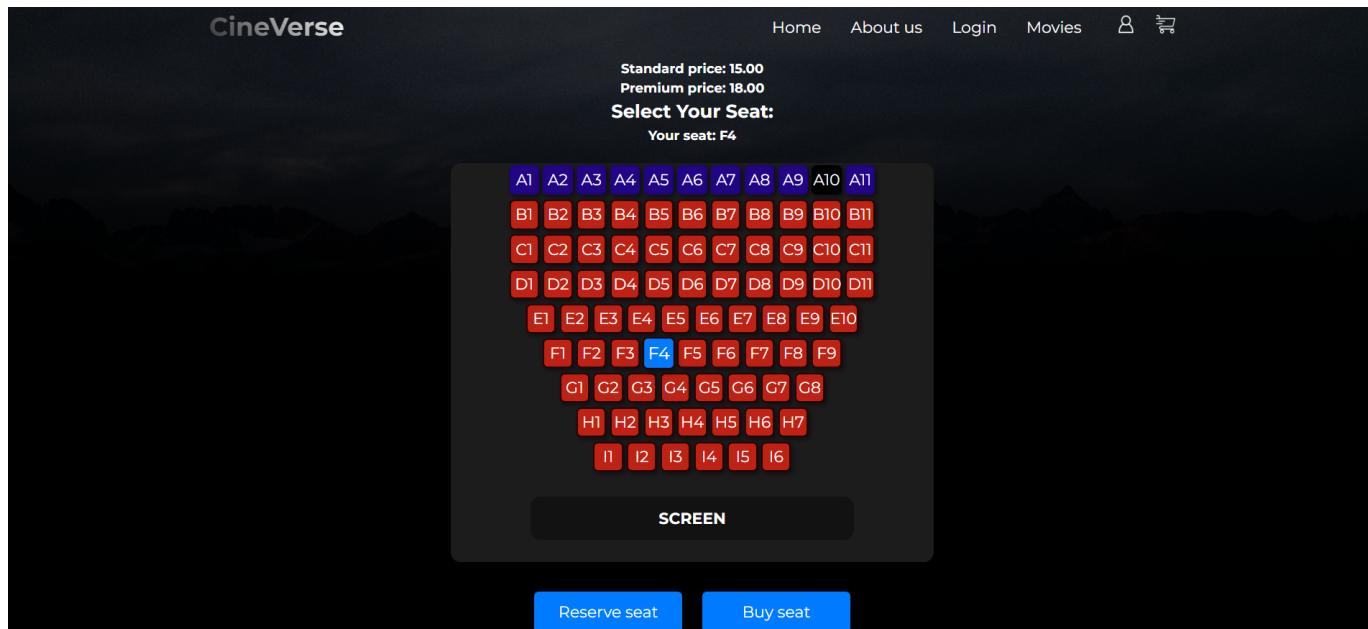
Standard price: 12.00 Home About us Login Movies

Select Your Seat:
Your seat: F4

Network response was not ok: 500 - Internal Server Error. Details: {\"error\": \"The movie screening must be later than 2 hours from the reservation time\\nCONTEXT: PL/pgSQL function check_reservation_period() line 12 at RAISE\\nSQL statement \"\\nINSERT INTO tickets (customerid, moviescreeningid, seatnumber, orderedondate, orderedontime, status)\\n VALUES (p_customer_id, p_movie_screening_id, p_seat_number, p_curr_date, p_curr_time, 'New')\\n\\nPL/pgSQL function reserve_movie_screening_seat(integer,integer,integer,date,time without time zone) line 8 at SQL statement\\n\"}"/>

SCREEN

Jak widzimy, nie da się zarezerwować tego miejsca, ponieważ seans zaczyna się 22.05.2024 o 14:30, a ustwiony systemowy czas to 22.05.2024, 13:00 (patrz punkt 2.10). Spróbujemy więc wybrać inny seans, to samo miejsce:



Widzimy, że miejsce już jest zajęte:

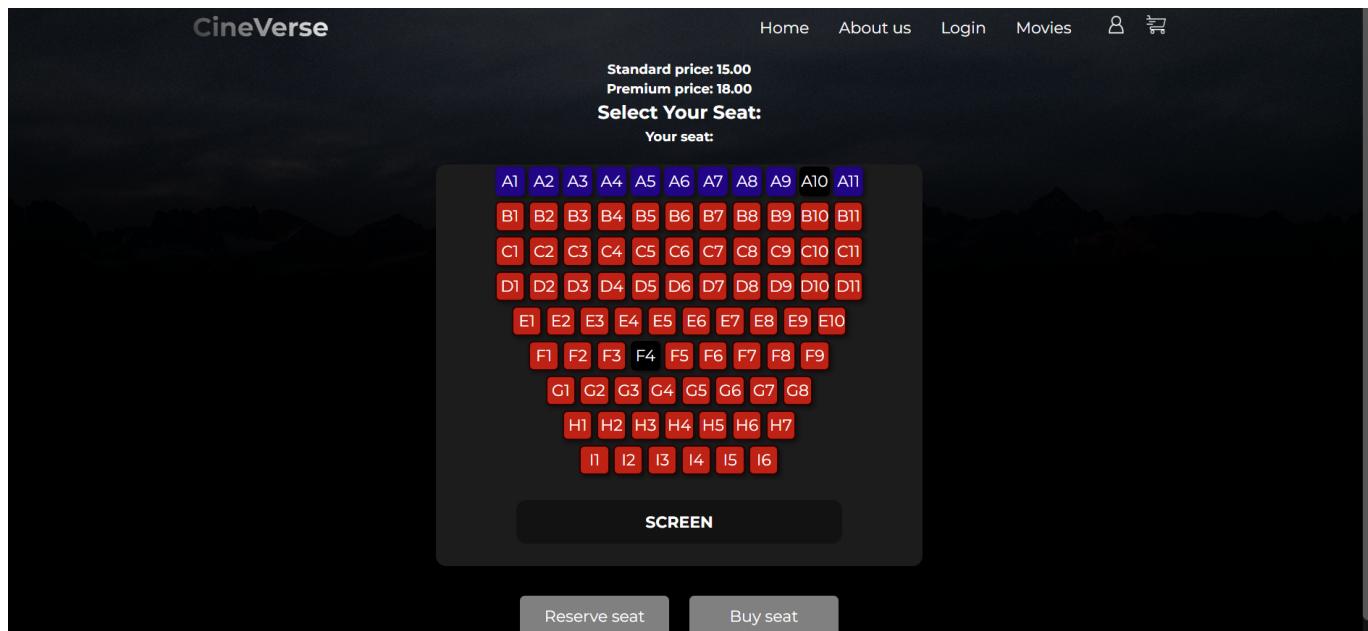


Tabela tickets po wywołaniu procedury:

	ticketid	customerid	orderedondate	orderedontime	status	moviescreeningid	seatnumber
1	27	6	2024-05-22	13:00:00	New	63	58
2	25	6	2024-05-21	16:30:00	Confirmed	64	10
3	24	6	2024-05-20	14:30:00	Confirmed	63	10
4	23	6	2024-05-22	13:00:00	Confirmed	15	60
5	22	6	2024-05-22	13:00:00	Confirmed	15	57
6	20	6	2024-05-22	13:00:00	Confirmed	15	79
7	19	6	2024-05-22	13:00:00	Canceled	22	81
8	17	6	2024-05-22	13:00:00	Confirmed	15	40
9	16	6	2024-05-22	13:00:00	Canceled	15	6
10	14	6	2024-05-22	13:00:00	Confirmed	1	6

- Zakup biletu

Tabela tickets przed wywołaniem procedury:

	ticketid	customerid	orderedondate	orderedontime	status	moviescreeningid	seatnumber
1		25	6 2024-05-21	16:30:00	Confirmed	64	10
2		24	6 2024-05-20	14:30:00	Confirmed	63	10
3		23	6 2024-05-22	13:00:00	Confirmed	15	60
4		22	6 2024-05-22	13:00:00	Confirmed	15	57
5		20	6 2024-05-22	13:00:00	Confirmed	15	79
6		19	6 2024-05-22	13:00:00	Canceled	22	81
7		17	6 2024-05-22	13:00:00	Confirmed	15	40
8		16	6 2024-05-22	13:00:00	Canceled	15	6
9		14	6 2024-05-22	13:00:00	Confirmed	1	6
10		13	6 2024-05-22	13:00:00	Confirmed	1	37

Wybrane przez nas miejsce A2, którego cena jest Premium:

Standard price: 15.00
Premium price: 18.00
Select Your Seat:
Your seat: A2

SCREEN

Reserve seat Buy seat

Widzimy, że miejsce już jest zajęte:

Standard price: 15.00
Premium price: 18.00
Select Your Seat:
Your seat:

SCREEN

Reserve seat Buy seat

Tabela tickets po wywołaniu procedury:

	ticketid	customerid	orderedondate	orderedontime	status	moviescreeningid	seatnumber
1	28	6	2024-05-22	13:00:00	Confirmed	63	2
2	27	6	2024-05-22	13:00:00	New	63	58
3	25	6	2024-05-21	16:30:00	Confirmed	64	10
4	24	6	2024-05-20	14:30:00	Confirmed	63	10
5	23	6	2024-05-22	13:00:00	Confirmed	15	60
6	22	6	2024-05-22	13:00:00	Confirmed	15	57
7	20	6	2024-05-22	13:00:00	Confirmed	15	79
8	19	6	2024-05-22	13:00:00	Canceled	22	81
9	17	6	2024-05-22	13:00:00	Confirmed	15	40
10	16	6	2024-05-22	13:00:00	Canceled	15	6

A teraz sprawdźmy poprawność wyliczania ceny dla tych biletów (Standard/Premium). W tym celu popatrzymy na informację wyświetlaną w profilu użytkownika:

Movie: Back to Black **Date:** 2024-05-28 **Start Time:** 12:20:00 **Duration:** 123 minutes **Hall Number:** 5 **Seat Number:** 58

Price: 15.00 **Status:** New **Ordered On:** 2024-05-22 at 13:00:00

Movie: Back to Black **Date:** 2024-05-28 **Start Time:** 12:20:00 **Duration:** 123 minutes **Hall Number:** 5 **Seat Number:** 10

Price: 18.00 **Status:** Confirmed **Ordered On:** 2024-05-20 at 14:30:00

Jak widać, cena biletu została policzona poprawnie.

- Zmiana statusu biletu

Spróbujmy kupić bilet, który jest zarezerwowany. W tym celu klikniemy odpowiedni przycisk w profilu użytkownika. Dostaliśmy wiadomość, że zmiana się powiodła:

Movie: IF **Date:** 2024-05-22 **Start Time:** 16:45:00 **Duration:** 107 minutes **Hall Number:** 3 **Seat Number:** 81

Price: 12.00 **Status:** Canceled **Ordered On:** 2024-05-22 at 13:00:00

Movie: Back to Black **Date:** 2024-05-28 **Start Time:** 12:20:00 **Duration:** 123 minutes **Hall Number:** 5 **Seat Number:** 2

Price: 18.00 **Status:** Confirmed **Ordered On:** 2024-05-22 at 13:00:00

Movie: Back to Black **Date:** 2024-05-28 **Start Time:** 12:20:00 **Duration:** 123 minutes **Hall Number:** 5 **Seat Number:** 10

Price: 15.00 **Status:** Confirmed **Ordered On:** 2024-05-20 at 14:30:00

Ticket status updated successfully!

Już teraz ten zarezerwowany ticket jest opłacony:

Movie: Back to Black	Date: 2024-05-28	Start Time: 12:20:00	Duration: 123 minutes	Hall Number: 5	Seat Number: 58
Price: 15.00 Status: Confirmed Ordered On: 2024-05-22 at 13:00:00					
Movie: Back to Black	Date: 2024-05-28	Start Time: 12:20:00	Duration: 123 minutes	Hall Number: 5	Seat Number: 10
Price: 18.00 Status: Confirmed Ordered On: 2024-05-20 at 14:30:00					

Admin

- Dodanie nowej kategorii

Tabela movie_categories przed wywołaniem procedury:

	moviecategoryid	categoryname
1	1	Action
2	2	Comedy
3	3	Drama
4	4	Thriller
5	5	Horror
6	6	Science Fiction
7	7	Biography
8	8	Romance
9	9	Animated

Wywołanie procedury z panelu admina na stronie:

MovieCategories

Add New Category

Delete Category

Add New Category

Category Name:

Confirm Cancel

MovieScreening

Add Week-Templated Movie Screenings

Delete Movie Screening

Tabela movie_categories po wywołaniu procedury:

	moviecategoryid	categoryname
1	1	Action
2	2	Comedy
3	3	Drama
4	4	Thriller
5	5	Horror
6	6	Science Fiction
7	7	Biography
8	8	Romance
9	9	Animated
10	56	TestCategory

- Usunięcie kategorii

Wywołanie procedury z panelu admina na stronie:

MovieCategories

Add New Category Delete Category

Delete Category

Category Name:

Confirm Cancel

MovieScreening

Add Week-Templated Movie Screenings Delete Movie Screening

Tabela movie_categories po wywołaniu procedury:

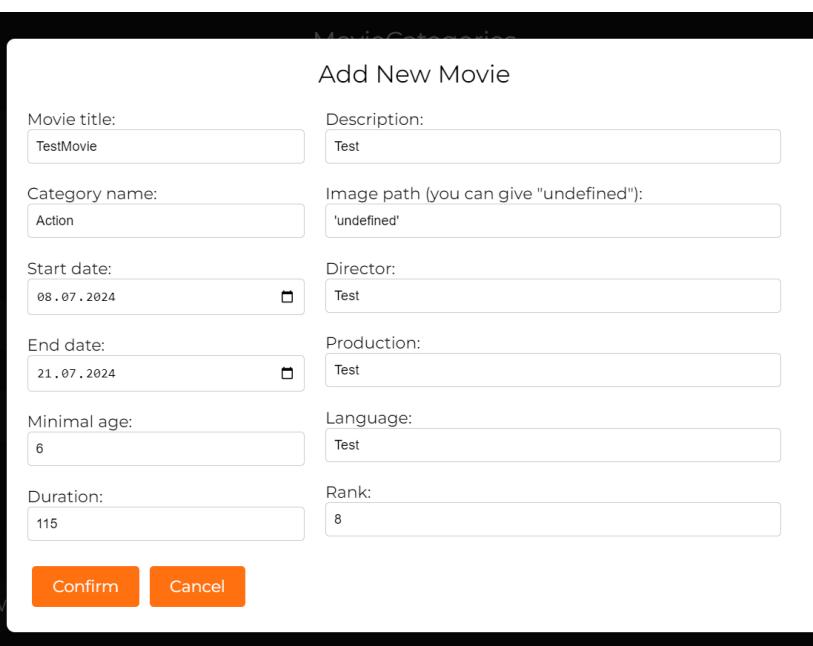
	moviecategoryid	categoryname
1	1	Action
2	2	Comedy
3	3	Drama
4	4	Thriller
5	5	Horror
6	6	Science Fiction
7	7	Biography
8	8	Romance
9	9	Animated

- Dodanie nowego filmu

Tabela movies przed wywołaniem procedury:

	movieid	moviecategoryid	title	startdate	enddate	duration	description
1	8	9	Garfield	2024-05-31	2024-06-15	101	Garfield is the most famous cat in the world.
2	9	9	Inside Out 2	2024-06-12	2024-06-28	108	Produced by Disney and Pixar, the movie is a sequel to the 2015 film.
3	10	9	Despicable Me 4	2024-07-05	2024-07-25	116	Gru, Lucy, Margo, Edith, and Agnes are back for another adventure.
4	1	2	IF	2024-05-17	2024-06-01	107	From writer and director John Krasinski, the movie explores the concept of what it means to be human.
5	2	8	Challengers	2024-04-26	2024-05-10	131	From visionary filmmaker Luca Guadagnino, the movie is a love story set in the 1970s.
6	4	9	Kung Fu Panda 4	2024-03-08	2024-03-22	94	This spring, for the first time in the franchise, Po's son, Meng, takes center stage.
7	5	7	Back to Black	2024-05-17	2024-05-31	123	A behind-the-scenes glimpse into the making of the iconic 2002 film.
8	6	1	Kingdom of the Planet of the Apes	2024-05-10	2024-05-24	145	Director Wes Ball breathes new life into the iconic franchise.
9	15	3	Dancing Queen	2024-06-07	2024-06-20	92	A 12-year-old girl who falls in love with a boy who can't dance.
10	11	1	Furiosa: A Mad Max Saga	2024-05-24	2024-06-06	148	The origin story of renegade warrior Furiosa.
11	3	5	Tarot	2024-05-03	2024-05-30	91	When a group of friends recklessly decide to play with fire.
12	12	3	Monster	2024-05-17	2024-06-12	126	Academy Award-nominated and Palme d'Or-winning film.
13	14	1	The Fall Guy	2024-05-03	2024-05-30	126	The film's hero is a stuntman, and he's back.

Wywołanie procedury z panelu admina na stronie:



The screenshot shows a modal dialog box titled "Add New Movie". The form contains the following fields:

- Movie title:
- Description:
- Category name:
- Image path (you can give "undefined"):
- Start date:
- Director:
- End date:
- Production:
- Minimal age:
- Language:
- Duration:
- Rank:

At the bottom of the modal are two buttons: "Confirm" and "Cancel".

Tabela movies po wywołaniu procedury:

	WHERE	ORDER BY	movieid	moviecategoryid	title	startdate	enddate	duration	description
1			8	9	Garfield	2024-05-31	2024-06-15	101	Garfield is the most fa
2			9	9	Inside Out 2	2024-06-12	2024-06-28	108	Produced by Disney and
3			10	9	Despicable Me 4	2024-07-05	2024-07-25	116	Gru, Lucy, Margo, Edith
4			1	2	IF	2024-05-17	2024-06-01	107	From writer and directo
5			2	8	Challengers	2024-04-26	2024-05-10	131	From visionary filmmake
6			4	9	Kung Fu Panda 4	2024-03-08	2024-03-22	94	This spring, for the fi
7			5	7	Back to Black	2024-05-17	2024-05-31	123	A behind-the-scenes gli
8			6	1	Kingdom of the Planet of the Apes	2024-05-10	2024-05-24	145	Director Wes Ball breat
9			15	3	Dancing Queen	2024-06-07	2024-06-20	92	A 12-year-old girl who
10			11	1	Furiosa: A Mad Max Saga	2024-05-24	2024-06-06	148	The origin story of ren
11			3	5	Tarot	2024-05-03	2024-05-30	91	When a group of friends
12			12	3	Monster	2024-05-17	2024-06-12	126	Academy Award-nominated
13			14	1	The Fall Guy	2024-05-03	2024-05-30	126	The film's hero is a st
14			16	1	Bad Boys Ride or Die	2024-06-07	2024-06-25	138	This Summer, the world\
15			7	1	Hit Man	2024-05-17	2024-06-01	115	A professor moonlightin
16			18	4	Force of Nature: The Dry 2	2024-05-17	2024-05-23	120	Five women participate
17			17	1	Turbo	2024-05-24	2024-06-12	110	Jose, a jeep driver fro
18			20	5	Amelia's children	2024-07-12	2024-08-01	91	When Edward's search fo
19			25	1	TEST	2024-06-08	2024-06-15	140	TEST
20			26	1	TEST2	2024-05-31	2024-06-13	81	TEST2
21			27	1	TestMovie	2024-07-08	2024-07-21	115	Test

Powstała nowa strona:

CineVerse

Home About us Login Movies

TestMovie

ACTION

Test

Start Date: 08.07.2024

End Date: 21.07.2024

Duration: 115 minutes

Director: Test

Minimum Age: 6

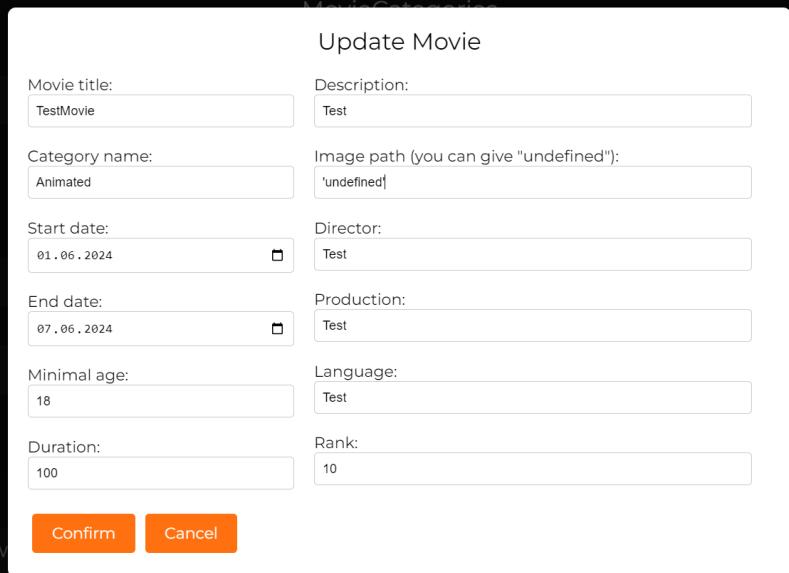
Production: Test

Original Language: Test

Buy Ticket

- Aktualizacja danych wybranego filmu

Wywołanie procedury z panelu admina na stronie:



MovieCategory

Update Movie

Movie title:

Description:

Category name:

Image path (you can give "undefined"):

Start date:

Director:

End date:

Production:

Minimal age:

Language:

Duration:

Rank:

Add Movie

Tabela movies po wywołaniu procedury:

	MovieId	MovieCategoryId	Title	Startdate	Enddate	Duration	Description
1	8	9	Garfield	2024-05-31	2024-06-15	101	Garfield is the most fa
2	9	9	Inside Out 2	2024-06-12	2024-06-28	108	Produced by Disney and I
3	10	9	Despicable Me 4	2024-07-05	2024-07-25	116	Gru, Lucy, Margo, Edith
4	1	2	IF	2024-05-17	2024-06-01	107	From writer and director
5	2	8	Challengers	2024-04-26	2024-05-10	131	From visionary filmmaker
6	4	9	Kung Fu Panda 4	2024-03-08	2024-03-22	94	This spring, for the fi
7	5	7	Back to Black	2024-05-17	2024-05-31	123	A behind-the-scenes gl
8	6	1	Kingdom of the Planet of the Apes	2024-05-10	2024-05-24	145	Director Wes Ball breath
9	15	3	Dancing Queen	2024-06-07	2024-06-20	92	A 12-year-old girl who
10	11	1	Furiosa: A Mad Max Saga	2024-05-24	2024-06-06	148	The origin story of ren
11	3	5	Tarot	2024-05-03	2024-05-30	91	When a group of friends
12	12	3	Monster	2024-05-17	2024-06-12	126	Academy Award-nominated
13	14	1	The Fall Guy	2024-05-03	2024-05-30	126	The film's hero is a st
14	16	1	Bad Boys Ride or Die	2024-06-07	2024-06-25	138	This Summer, the world\
15	7	1	Hit Man	2024-05-17	2024-06-01	115	A professor moonlightin
16	18	4	Force of Nature: The Dry 2	2024-05-17	2024-05-23	120	Five women participate
17	17	1	Turbo	2024-05-24	2024-06-12	110	Jose, a jeep driver fro
18	20	5	Amelia's children	2024-07-12	2024-08-01	91	When Edward's search fo
19	25	1	TEST	2024-06-08	2024-06-15	140	TEST
20	26	1	TEST2	2024-05-31	2024-06-13	81	TEST2
21	27	9	TestMovie	2024-06-01	2024-06-07	100	Test

- Usunięcie filmu

Wywołanie procedury z panelu admina na stronie:

MovieCategories

Add New Category Delete Category

Delete Movie

Movie Name:

Confirm Cancel

MovieScreening

Add Week-Templated Movie Screenings Delete Movie Screening

Tabela movies po wywołaniu procedury:

	movieid	moviecategoryid	title	startdate	enddate	duration	description
1	8	9	Garfield	2024-05-31	2024-06-15	101	Garfield is the most fa
2	9	9	Inside Out 2	2024-06-12	2024-06-28	108	Produced by Disney and I
3	10	9	Despicable Me 4	2024-07-05	2024-07-25	116	Gru, Lucy, Margo, Edith
4	1	2	IF	2024-05-17	2024-06-01	107	From writer and director
5	2	8	Challengers	2024-04-26	2024-05-10	131	From visionary filmmaker
6	4	9	Kung Fu Panda 4	2024-03-08	2024-03-22	94	This spring, for the fi
7	5	7	Back to Black	2024-05-17	2024-05-31	123	A behind-the-scenes glin
8	6	1	Kingdom of the Planet of the Apes	2024-05-10	2024-05-24	145	Director Wes Ball breath
9	15	3	Dancing Queen	2024-06-07	2024-06-20	92	A 12-year-old girl who
10	11	1	Furiosa: A Mad Max Saga	2024-05-24	2024-06-06	148	The origin story of ren
11	3	5	Tarot	2024-05-03	2024-05-30	91	When a group of friends
12	12	3	Monster	2024-05-17	2024-06-12	126	Academy Award-nominated
13	14	1	The Fall Guy	2024-05-03	2024-05-30	126	The film's hero is a stu
14	16	1	Bad Boys Ride or Die	2024-06-07	2024-06-25	138	This Summer, the world\
15	7	1	Hit Man	2024-05-17	2024-06-01	115	A professor moonlightin
16	18	4	Force of Nature: The Dry 2	2024-05-17	2024-05-23	120	Five women participate
17	17	1	Turbo	2024-05-24	2024-06-12	110	Jose, a jeep driver from
18	20	5	Amelia's children	2024-07-12	2024-08-01	91	When Edward's search for
19	25	1	TEST	2024-06-08	2024-06-15	140	TEST
20	26	1	TEST2	2024-05-31	2024-06-13	81	TEST2

- Dodanie seansów na określoną liczbę dni

Tabela movie_screening przed wywołaniem procedury:

	moviescreeningid	movieid	date	starttime	endtime	pricestandard	pricepremium	threedimensional
1	213	25	2024-06-10	13:00:00	15:20:00	18.00	21.00	• true
2	212	25	2024-06-10	14:00:00	16:20:00	10.00	11.00	• true
3	211	25	2024-06-09	13:00:00	15:20:00	18.00	21.00	• true
4	209	17	2024-05-28	10:00:00	11:50:00	12.00	15.00	false
5	208	17	2024-05-27	10:00:00	11:50:00	12.00	15.00	false
6	207	17	2024-05-26	10:00:00	11:50:00	12.00	15.00	false
7	206	17	2024-05-25	10:00:00	11:50:00	12.00	15.00	false
8	205	17	2024-05-24	10:00:00	11:50:00	12.00	15.00	false
9	204	14	2024-05-28	21:30:00	23:36:00	18.00	21.00	false
10	203	14	2024-05-27	21:30:00	23:36:00	18.00	21.00	false
11	202	14	2024-05-26	21:30:00	23:36:00	18.00	21.00	false
12	201	14	2024-05-25	21:30:00	23:36:00	18.00	21.00	false
13	200	14	2024-05-24	21:30:00	23:36:00	18.00	21.00	false

Spróbujemy dodać seansy na 7 dni, zaczynając od 11-06-2024, mimo że ostatni dzień kiedy film jest grany to 15-06-2024. Wywołanie procedury z panelu admina na stronie:

Add Week-Templated Movie Screenings

Movie title:	Price standard:
TEST	16
Date:	Price premium:
11.06.2024	20
Start time:	Language:
18:00	English
Is 3D?:	Movie hall:
<input checked="" type="checkbox"/>	6
How much days since given date?	
7	

Confirm

Cancel

Network response was not ok: 500 - Internal Server Error. Details: {"error": "Movie with title TEST does not exist or is not available on the specified date\nCONTEXT: PL/pgSQL function add_movie_screening(character varying,date,time without time zone,numeric,numeric,boolean,character varying,integer) line 16 at RAISE\nSQL statement\n\"CALL add_movie_screening(\r\n movie_title,\r\n current_date_val,\r\n start_time,\r\n price_standard,\r\n price_premium,\r\n is_3d,\r\n language_val,\r\n hall_number\r\n)\"\nPL/pgSQL function add_movie_screenings_weekly(character varying,date,time without time zone,numeric,numeric,boolean,character varying,integer,integer) line 24 at CALL\n"}

Jak widać, dostaliśmy błąd. Sprawdźmy, czy zostały utrwalone seansy dla terminów, które są poprawne. Tabela movie_screening po wywołaniu procedury:

	moviescreeningid	movieid	date	starttime	endtime	pricestandard	pricepremium	threedimensional
1	213	25	2024-06-10	13:00:00	15:20:00	18.00	21.00	• true
2	212	25	2024-06-10	14:00:00	16:20:00	10.00	11.00	• true
3	211	25	2024-06-09	13:00:00	15:20:00	18.00	21.00	• true
4	209	17	2024-05-28	10:00:00	11:50:00	12.00	15.00	false
5	208	17	2024-05-27	10:00:00	11:50:00	12.00	15.00	false
6	207	17	2024-05-26	10:00:00	11:50:00	12.00	15.00	false
7	206	17	2024-05-25	10:00:00	11:50:00	12.00	15.00	false
8	205	17	2024-05-24	10:00:00	11:50:00	12.00	15.00	false
9	204	14	2024-05-28	21:30:00	23:36:00	18.00	21.00	false
10	203	14	2024-05-27	21:30:00	23:36:00	18.00	21.00	false
11	202	14	2024-05-26	21:30:00	23:36:00	18.00	21.00	false
12	201	14	2024-05-25	21:30:00	23:36:00	18.00	21.00	false
13	200	14	2024-05-24	21:30:00	23:36:00	18.00	21.00	false

Jak widać, błąd przerwał całą transakcję i poprawne dane nie zostały dodane. Spróbujmy zmienić liczbę dni na zgodne z zasadami:

Add Week-Templated Movie Screenings

Movie title:

Price standard:

Date:

Price premium:

Start time:

Language:

Is 3D?:

Movie hall:

How much days since given date?

Week-Templated MovieScreening were added successfully!

Po wpisywaniu liczby dni równej 4, dostaliśmy komunikat, że udało się dodać te dane. Tabela movie_screening po wywołaniu procedury:

	movieScreeningId	movieId	date	startTime	endTime	priceStandard	pricePremium	threeDimensional
1	227	25	2024-06-15	18:00:00	20:20:00	16.00	20.00	• true
2	226	25	2024-06-14	18:00:00	20:20:00	16.00	20.00	• true
3	225	25	2024-06-13	18:00:00	20:20:00	16.00	20.00	• true
4	224	25	2024-06-12	18:00:00	20:20:00	16.00	20.00	• true
5	223	25	2024-06-11	18:00:00	20:20:00	16.00	20.00	• true
6	213	25	2024-06-10	13:00:00	15:20:00	18.00	21.00	• true
7	212	25	2024-06-10	14:00:00	16:20:00	10.00	11.00	• true
8	211	25	2024-06-09	13:00:00	15:20:00	18.00	21.00	• true
9	209	17	2024-05-28	10:00:00	11:50:00	12.00	15.00	false
10	208	17	2024-05-27	10:00:00	11:50:00	12.00	15.00	false
11	207	17	2024-05-26	10:00:00	11:50:00	12.00	15.00	false
12	206	17	2024-05-25	10:00:00	11:50:00	12.00	15.00	false
13	205	17	2024-05-24	10:00:00	11:50:00	12.00	15.00	false

- Usunięcie wybranego seansu

Wywołanie procedury z panelu admina na stronie:

MovieCategories

Add New Category Delete Category

Delete Movie Screening

Movie title: Date: Start time:

Confirm Cancel

MovieScreening

Add Week-Templated Movie Screenings Delete Movie Screening

Tabela movie_screening po wywołaniu procedury:

	moviescreeningid	movieid	date	starttime	endtime	pricestandard	pricepremium	threedimensional
1	226	25	2024-06-14	18:00:00	20:20:00	16.00	20.00	• true
2	225	25	2024-06-13	18:00:00	20:20:00	16.00	20.00	• true
3	224	25	2024-06-12	18:00:00	20:20:00	16.00	20.00	• true
4	223	25	2024-06-11	18:00:00	20:20:00	16.00	20.00	• true
5	213	25	2024-06-10	13:00:00	15:20:00	18.00	21.00	• true
6	212	25	2024-06-10	14:00:00	16:20:00	10.00	11.00	• true
7	211	25	2024-06-09	13:00:00	15:20:00	18.00	21.00	• true
8	209	17	2024-05-28	10:00:00	11:50:00	12.00	15.00	false
9	208	17	2024-05-27	10:00:00	11:50:00	12.00	15.00	false
10	207	17	2024-05-26	10:00:00	11:50:00	12.00	15.00	false