

Politechnika Łódzka

Zaawansowane systemy baz danych

Projekt końcowy - 'SportCenter'

Krótki opis wykonanego zadania

*Michał Markiewicz
Dawid Wolszczak*

1. Purpose of the project

Nasz projekt zakładał stworzenie systemu rezerwacji dla firmy, która świadczy usługi w zakresie udostępniania osobom prywatnym terenów do uprawiania sportu (takich jak boiska do piłki nożnej czy koszykówki).

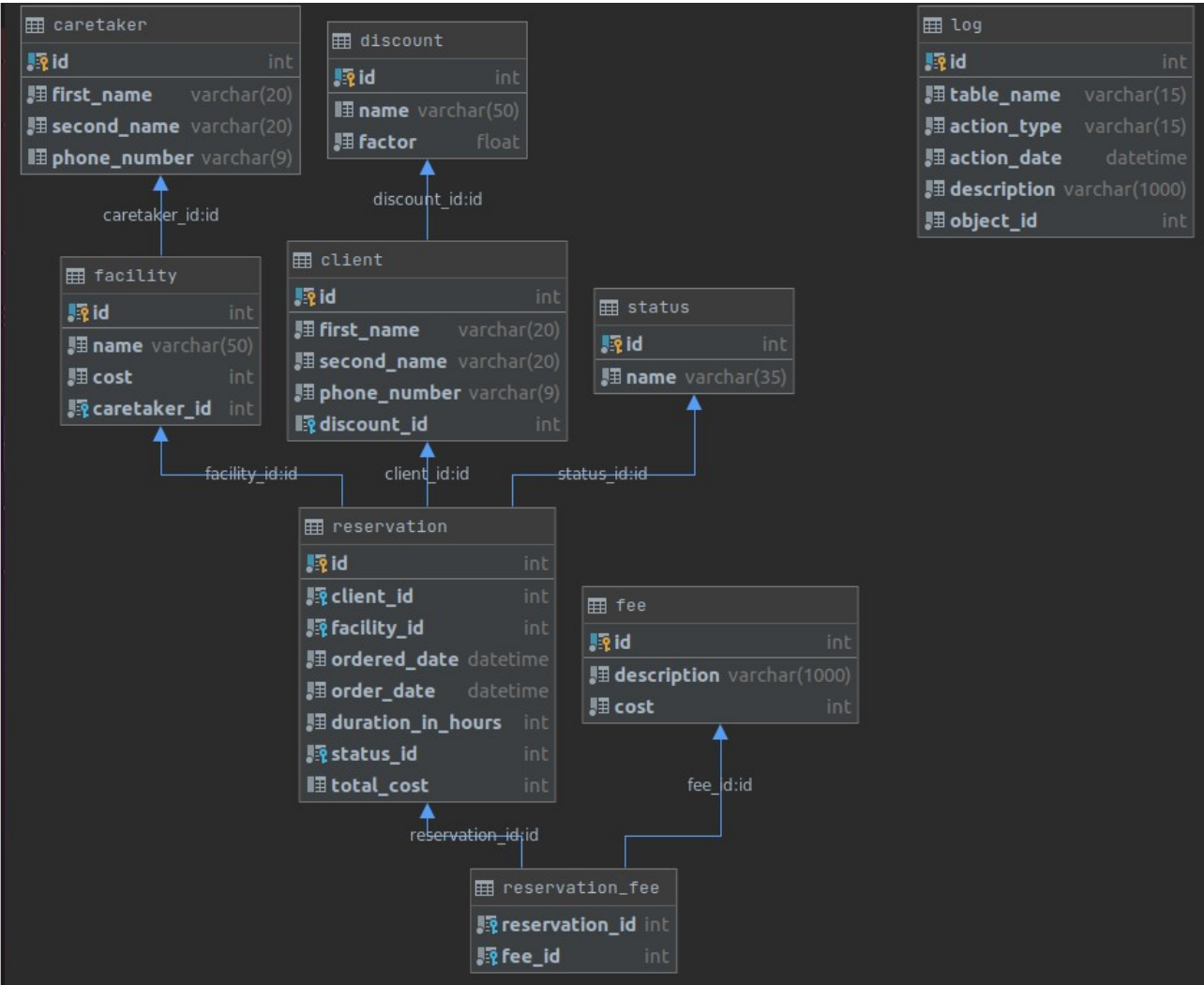
2. Project content

Poza plikami .sql zawierającymi model bazy danych, deklaracje funkcji, wyzwalaczy i procedur oraz wstępne dane - stworzyliśmy w formie przeglądarkowej aplikację kliencką, która za pośrednictwem serwera backend'owego pozwala na modyfikację danych w bazie w czasie rzeczywistym.

3. Technologies used in project

- dialekt SQL → T-SQL
- serwer SQL → MSSQL
- backend → Python/Flask
- frontend → Vue.js
- request sender → axios.js

4. Database scheme



5. Database system content

I. Tabele:

- client
- fee
- discount
- caretaker
- reservation
- reservation_fee
- status
- logs
- facility

II. Procedury (czy wynik widoczny na frontendzie)

- create_client tak
- update_client tak
- delete_client nie widoczny
- create_fee tak
- update_fee tak
- delete_fee nie widoczny
- create_facility tak
- update_facility tak
- delete_facility nie widoczny
- create_caretaker tak
- update_caretaker tak
- delete_caretaker nie widoczny
- create_discount tak
- update_discount tak
- delete_discount tak
- create_reservation tak
- update_reservation tak
- complete_reservation tak
- delete_reservation tak
- add_fee_to_reservation tak
- delete_fee_from_reservation nie widoczny

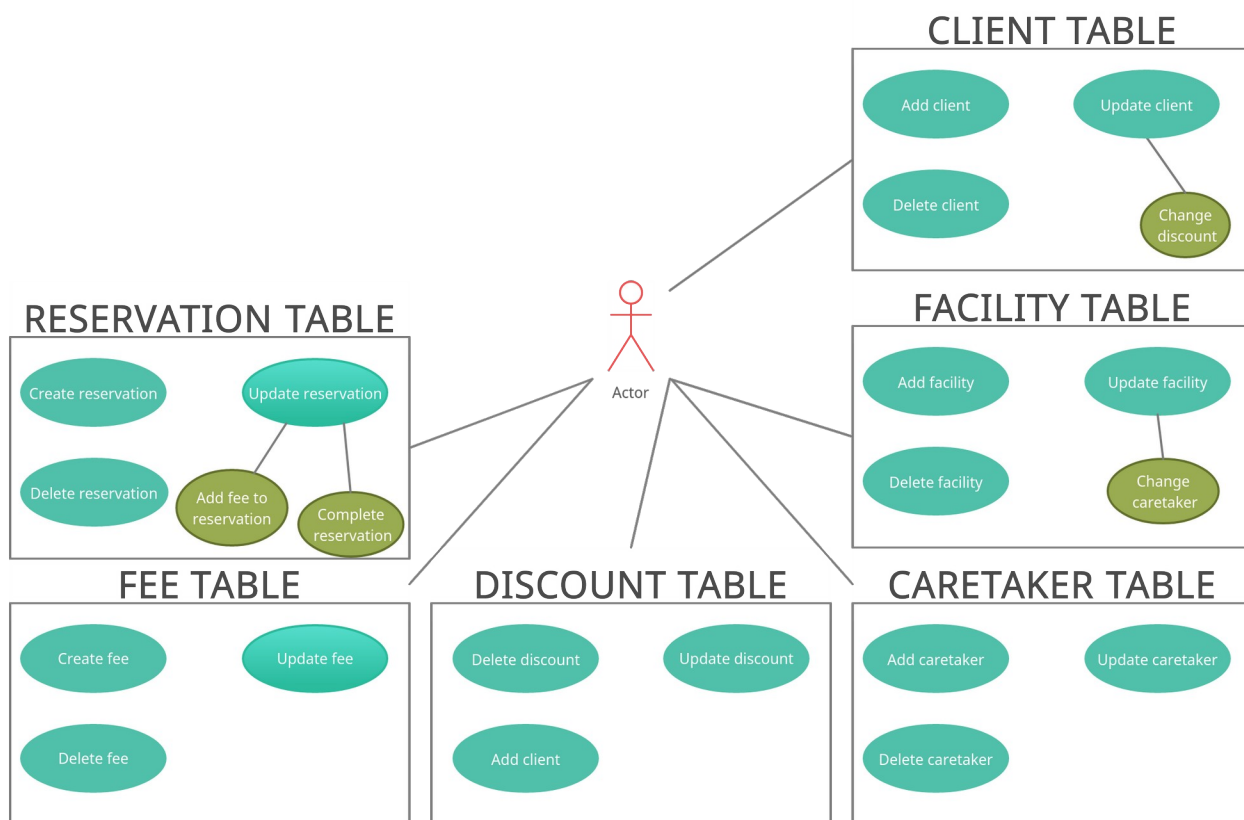
III. Funkcje

- get_client_list tak
- get_fee_list tak
- get_facility_list tak
- get_caretaker_list tak
- get_discount_list tak
- get_reservation_list tak
- get_log_list tak
- get_status_list tak
- get_reservations_for_this_day_and_facility tak
- check_reservation_availability tak
- get_reservations_for_client tak
- calculate_total_price tak
- complete_reservation tak
- get_fee_list_for_reservation tak

IV. Wyzwalacze

- trigger_discount_insert
- trigger_client_insert
- trigger_fee_insert
- trigger_status_insert
- trigger_caretaker_insert
- trigger_facility_insert
- trigger_reservation_insert
- trigger_reservation_fee_insert
- trigger_discount_update
- trigger_client_update
- trigger_fee_update
- trigger_status_update
- trigger_caretaker_update
- trigger_facility_update
- trigger_reservation_update

6. Database manipulation cases



7. API routes

```
@app.route('/client/<client_id>/getreservationlist', methods=['GET'])
@app.route('/client/getlist', methods=['GET'])
@app.route('/client/create', methods=['POST'])
@app.route('/client/<client_id>/update', methods=['PUT'])
def client(client_id=None):...

@app.route('/discount/create', methods=['POST'])
@app.route('/discount/getlist', methods=['GET'])
@app.route('/discount/<discount_id>/update', methods=['PUT'])
def discount(discount_id=None):...

@app.route('/fee/create', methods=['POST'])
@app.route('/fee/getlist', methods=['GET'])
@app.route('/fee/<fee_id>/update', methods=['PUT'])
def fee(fee_id=None):...
```

```
@app.route('/facility/create', methods=['POST'])
@app.route('/facility/<facility_id>/update', methods=['PUT'])
@app.route('/facility/getlist', methods=['GET'])
def facility(facility_id=None):...

@app.route('/caretaker/<caretaker_id>/update', methods=['PUT'])
@app.route('/caretaker/create', methods=['POST'])
@app.route('/caretaker/getlist', methods=['GET'])
def caretaker(caretaker_id=None):...
```

```
@app.route('/reservation/create', methods=['POST'])
@app.route('/reservation/getlist', methods=['GET'])
@app.route('/reservation/check', methods=['POST'])
@app.route('/reservation/check/exclude/<reservation_id>', methods=['POST'])
@app.route('/reservation/<reservation_id>/update', methods=['PUT'])
@app.route('/reservation/<reservation_id>/getfeelist', methods=['GET'])
@app.route('/reservation/<reservation_id>/addfee', methods=['POST'])
@app.route('/reservation/<reservation_id>/complete', methods=['GET'])
def reservation(reservation_id=0):...

@app.route('/log/getlist/<sort_by>', methods=['GET'])
@app.route('/log/getlist', methods=['GET'])
def log(sort_by=None):...
```

8. Sample views from client web application

- tworzenie rezerwacji wraz z tabelą wszystkich rezerwacji

The screenshot shows the SportCenter web application interface. On the left is a sidebar with various icons. The main content area is divided into two parts. The left part contains a reservation form with the following fields:

- Client name: David Wolszczak (dropdown menu)
- Filter clients: (text input)
- Facility name: Football grass field (dropdown menu)
- Filter clients: (text input)
- Order date: 01/04/2021 (text input)
- Order time: 06:30 (text input)
- Duration in hours: 2 (text input)
- Buttons: CHECK THE AVAILABILITY OF THE DATE, SUBMIT

The right part displays a table of reservations:

ID	Total cost(or update complete)	Update	Client ID	Facility ID	Booked date	Order date	Order duration	Status	Fees
1	440	U	1	1	Tue, 29 Dec 2020 02:50:26 GMT	Wed, 30 Dec 2020 07:00:00 GMT	2	4	S
2	COM	U	2	1	Tue, 29 Dec 2020 02:50:26 GMT	Wed, 30 Dec 2020 10:00:00 GMT	1	1	S
3	COM	U	3	1	Tue, 29 Dec 2020 02:50:26 GMT	Wed, 30 Dec 2020 15:00:00 GMT	5	1	S

- tabela logów z bazy danych

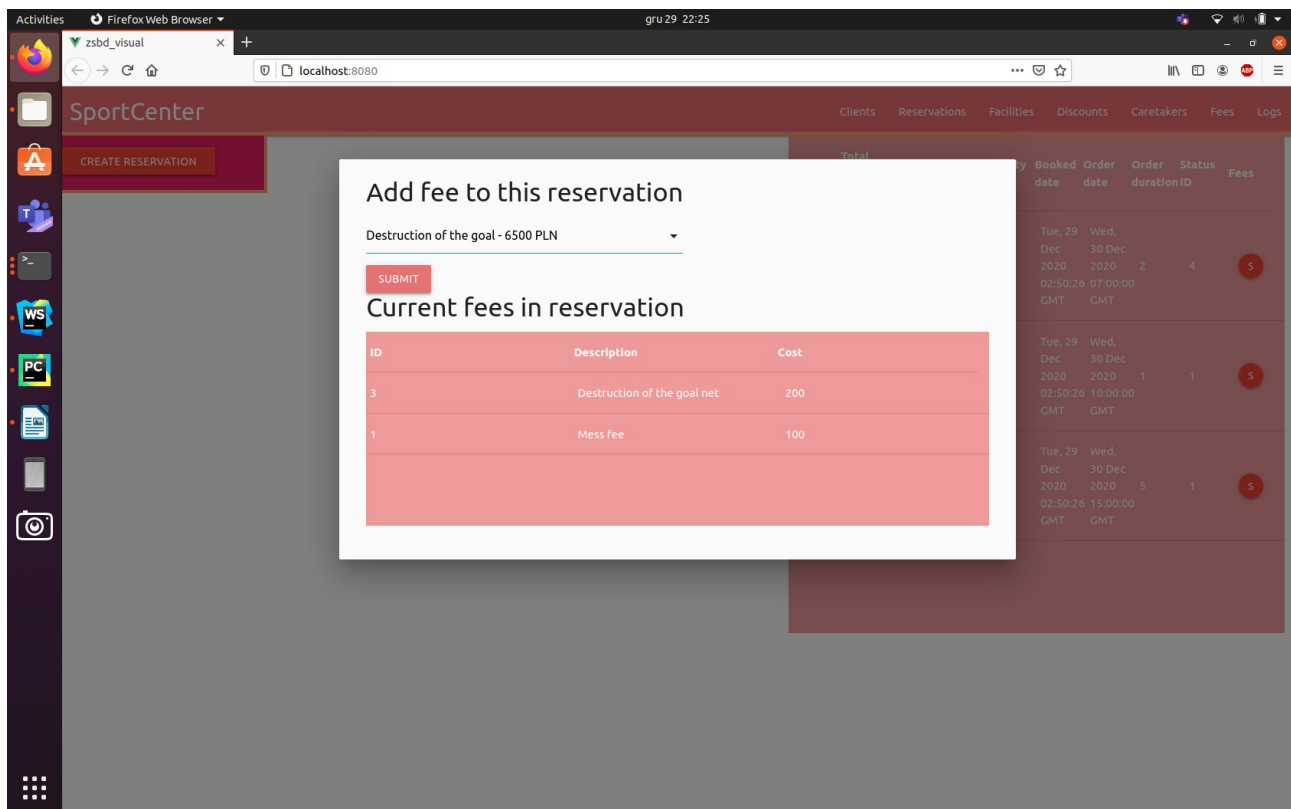
The screenshot shows the SportCenter web application interface. The main content area displays a table of database logs:

ID	Table name	Action type	Action date	Table object ID	Description
1	discount	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	4	Discount "Top customer" has been created
2	discount	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	3	Discount "Golden customer" has been created
3	discount	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	2	Discount "Regular customer" has been created
4	discount	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	1	Discount "More than one" has been created
5	client	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	4	Client "Zbigniew Stonoga" has been inserted
6	client	insert(create)	Tue, 29 Dec 2020 00:00:00 GMT	3	Client "Donald Tusk" has been inserted

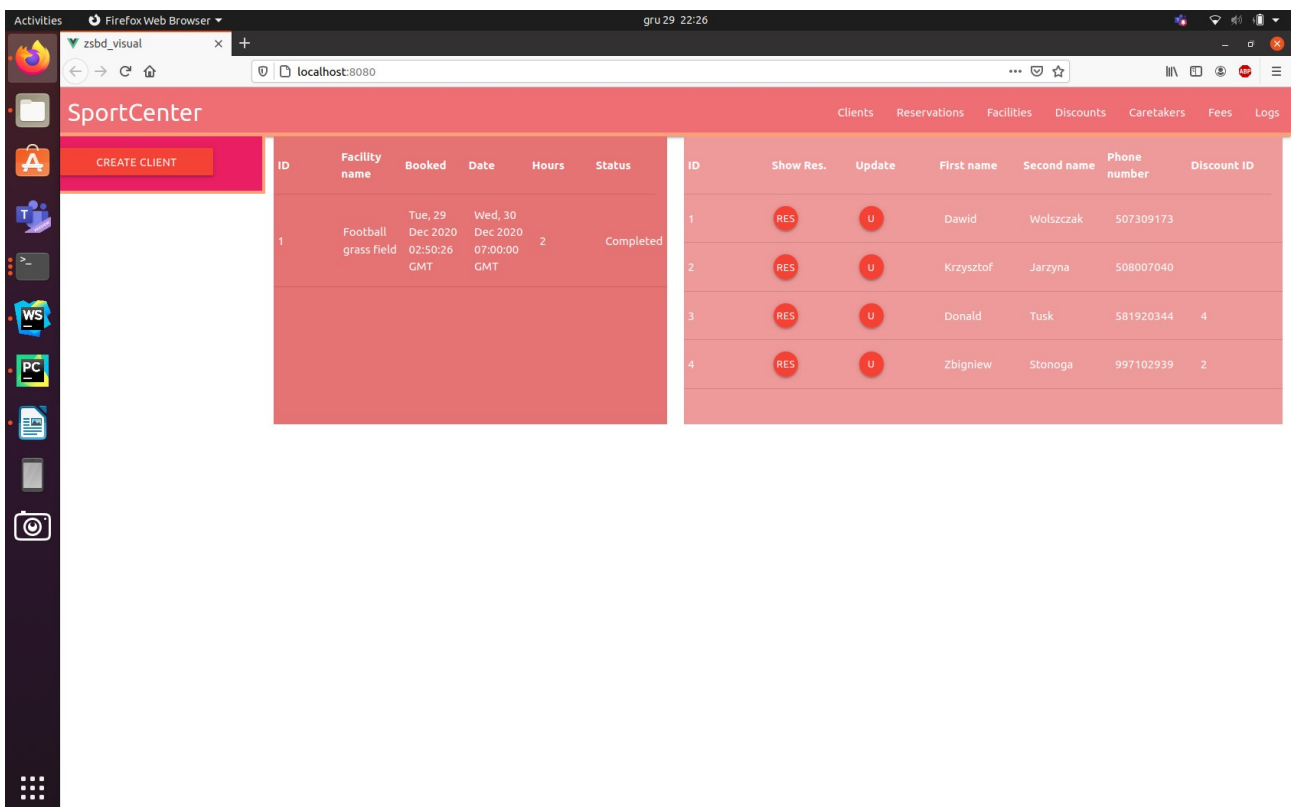
Below the table, there are four buttons for sorting the logs:

- DEFAULT SORT BY ID
- SORT BY TABLE NAME
- SORT BY ACTION TYPE
- SORT BY ACTION DATE

- dodawanie dodatkowych opłat dla danej rezerwacji wraz z tabelą aktualnych opłat



- po prawej tabela wyświetlająca wszystkie rezerwacje danego użytkownika, po lewej tabela wszystkich użytkowników



9. Client-server example communication

- tworzenie użytkownika*

method: POST

route: /reservation/create,

```
request.data = {  
  first_name: VARCHAR(32)  
  second_name VARCHAR(32),  
  phone_number VARCHAR(9),  
}
```

response.data = 'OK'

lub

response.data = 'FAILURE'

response.status = 200

- pobieranie listy użytkowników*

method: GET

route: client/getlist

response.data

