

IPW

UAIM

Laboratorium 4

Karol Żelazowski 324953

Spis treści

1. Implementacja endpoint'ów	3
1.1. Teacher-list	3
1.2. Teacher-datails	3
1.3. Book-lesson	4
1.4. Add-teacher	4
1.5. Get-lessons	5
2. Implementacja żądań	6
2.1. Ogólna budowa żądania	6
2.2. Żądanie teacher-list	6
2.3. Zapytanie teacher-details	9
2.3.1. Test z poprawnym parametrem	9
2.3.2. Test z złym parametrem	11
2.4. Zapytanie book-lessons	13
2.4.1. Test z poprawnym terminem	13
2.4.2. Test z zajęтым terminem	14
2.4.3. Test ze złym terminem	16
2.5. Zapytanie add-teacher	18
2.5.1. Test z poprawnymi danymi	18
2.5.2. Test ze złymi danymi	20
2.6. Zapytanie get-lessons	22
2.6.1. Test z poprawnymi parametrami	22
2.6.2. Test ze złym parametrem	25

1. Implementacja endpoint'ów

1.1. Teacher-list

Wyciąga z bazy danych listę nauczycieli i prezentuje ją za pomocą obiektu JSON

```
@app.route('/teacher-list', methods=['GET'])
def get_teacher_list():
    teachers = db.session.query(Teacher).all()
    response = [
        {
            "id_nauczyciela": teacher.id_nauczyciela,
            "imie": teacher.imie,
            "nazwisko": teacher.nazwisko,
            "prowadzone_przedmioty": teacher.prowadzone_przedmioty
        } for teacher in teachers
    ]
    return jsonify(response), 200
```

Rysunek 1: Implementacja endpoint'u teacher-list

1.2. Teacher-details

Endpoint przyjmuje argument id_nauczyciela. Następnie zwraca dane nauczyciela o takim id

```
@app.route('/teacher-details', methods=['GET'])
def get_teacher_details():
    id_nauczyciela = request.args.get('id_nauczyciela')
    nauczyciel = db.session.query(Teacher).filter(Teacher.id_nauczyciela == id_nauczyciela).first()
    if not nauczyciel:
        return jsonify({"error": "Nauczyciel nie istnieje"}), 404

    result = {
        "id_nauczyciela": nauczyciel.id_nauczyciela,
        "imie": nauczyciel.imie,
        "nazwisko": nauczyciel.nazwisko,
        "prowadzone_przedmioty": nauczyciel.prowadzone_przedmioty,
        "opis": nauczyciel.opis,
        "ocena_nauczyciela": nauczyciel.ocena_nauczyciela,
        "numer_telefonu": nauczyciel.numer_telefonu,
        "stawka": nauczyciel.stawka,
        "waluta": nauczyciel.waluta,
        "email": nauczyciel.email,
        "dostepnosc": [
            {
                "dostepny_od": x.dostepny_od.strftime('%H:%M:%S'),
                "dostepny_do": x.dostepny_do.strftime('%H:%M:%S'),
            } for x in nauczyciel.dostepnosc
        ]
    }
    return jsonify(result), 200
```

Rysunek 2: Implementacja endpoint'u teacher-detail

1.3. Book-lesson

Endpoint odpowiedzialny za dodanie lekcji do bazy danych. Endpoint przyjmuje obiekt JSON z informacjami na temat lekcji. Jeśli podany nauczyciel ma już lekcje w danym terminie lub nie udziela wtedy lekcji, endpoint zwraca odpowiednią informację, że nie udało się zarejestrować lekcji. Natomiast jeśli rezerwacja przejdzie pomyślnie zwracana jest odpowiednia informacja. W przypadku błędu w trakcie dodawania lekcji do bazy danych użytkownik jest o tym informowany.

```
@app.route('/book-lesson', methods=['POST'])
def book_lesson():
    data = request.get_json()
    id_studenta = data.get('id_studenta')
    id_nauczyciela = data.get('id_nauczyciela')
    nazwa_przedmiotu = data.get('nazwa_przedmiotu')
    termin = data.get('termin')

    student = db.session.query(Student).filter(Student.id_studenta == id_studenta).first()
    nauczyciel = db.session.query(Teacher).filter(Teacher.id_nauczyciela == id_nauczyciela).first()
    przedmiot = db.session.query(SubjectsList).filter(SubjectsList.nazwa_przedmiotu == nazwa_przedmiotu).first()
    dostepnosc = db.session.query(TeacherCalendar).filter(TeacherCalendar.id_nauczyciela == id_nauczyciela).first()
    termin_date = datetime.strptime(termin, "%Y-%m-%dT%H:%M:%S")

    if termin_date.time() < dostepnosc.dostepny_od or termin_date.time() > dostepnosc.dostepny_do:
        return jsonify({"error": "Nauczyciel nie jest dostępny w tych godzinach"}), 400

    other_lessons = db.session.query(Lesson).filter(Lesson.id_nauczyciela == id_nauczyciela, Lesson.data_lekcji == termin_date.date()).all()
    for lesson in other_lessons:
        if lesson.data_lekcji == termin_date.date():
            return jsonify({"error": "Nauczyciel jest już zajęty w podanym terminie!"}), 400
    try:
        new_lesson = Lesson(
            id_nauczyciela = id_nauczyciela,
            id_studenta = id_studenta,
            id_przedmiotu = przedmiot.id_przedmiotu,
            data_lekcji = termin_date.date()
        )
        db.session.add(new_lesson)
        db.session.commit()
        return jsonify({"message": "Lekcja została zarezerwowana"}), 200
    except Exception as e:
        return jsonify({"error": "Nie udało się zarezerwować lekcji"}), 400
```

Rysunek 3: Implementacja endpoint'u book-lesson

1.4. Add-teacher

Endpoint odpowiedzialny za dodanie nauczyciela do bazy danych przyjmuje obiekt JSON z danymi na temat nauczyciela. Jeśli nie zostaną podane wszystkie informacje na temat nauczyciela endpoint zwróci błąd. W przypadku poprawnego dodania, użytkownik zostanie o tym poinformowany.

```

@app.route('/add-teacher', methods=['POST'])
def add_teacher():
    data = request.get_json()
    try:
        new_teacher = Teacher(
            imie = data['imie'],
            nazwisko = data['nazwisko'],
            prowadzone_przedmioty = data['prowadzone_przedmioty'],
            opis = data.get('opis'),
            ocena_nauczyciela = data['ocena_nauczyciela'],
            numer_telefonu = data['numer_telefonu'],
            stawka = data['stawka'],
            waluta = data['waluta'],
            email = data['email']
        )
        db.session.add(new_teacher)
        db.session.commit()
        return jsonify({"id_nauczyciela": new_teacher.id_nauczyciela}), 200
    except KeyError as ke:
        # Błąd związany z brakiem klucza w danych wejściowych
        return jsonify({"error": f"Brak wymaganej wartości: {str(ke)}"}), 400
    except Exception as e:
        # Inne błędy (np. problemy z bazą danych)
        print(f"Błąd: {str(e)}") # Wypisanie błędu do konsoli (logowanie)
        return jsonify({"error": f"Wystąpił błąd: {str(e)}"}), 500

```

Rysunek 4: Implementacja endpoint'u add-teacher

1.5. Get-lessons

Endpoint odpowiedzialny za zwrócenie listy lekcji, w których uczestniczył dany uczeń w podanym okresie. Dane wejściowe są wprowadzane za pomocą obiektu JSON

```

@app.route('/get-lessons', methods=['GET'])
def lessons_by_student_and_date():
    id_studenta = request.args.get('id_studenta')
    data_od = request.args.get('data_od')
    data_do = request.args.get('data_do')

    student = db.session.query(Student).filter(Student.id_studenta == id_studenta).first()
    if not student:
        return jsonify({"error": "Student nie istnieje"}), 404

    lessons = db.session.query(Lesson).filter(Lesson.id_studenta == id_studenta, Lesson.data_lekcji.between(data_od, data_do)).all()

    response = []
    for lesson in lessons:
        {
            "data_lekcji" : lesson.data_lekcji,
            "id_nauczyciela" : lesson.id_nauczyciela,
            "imie_nauczyciela" : lesson.teacher.imie,
            "nazwisko_nauczyciela" : lesson.teacher.nazwisko,
            "przedmiot" : lesson.subject.nazwa_przedmiotu
        }
    return jsonify(response), 200

```

Rysunek 5: Implementacja endpoint'u get-lessons

2. Implementacja żądań

2.1. Ogólna budowa żądania

Bazowe URL podana jest jako zmienna, której nie da się zmienić. Tak samo jak HEADERS, która zawiera nagłówek autoryzacyjny. Funkcja budująca żądania przyjmuje cztery argumenty.

- endpoint - odpowiedzialny do kierowania zapytania na odpowiedni endpoint
- method - określający metodę zapytania, czy GET, czy POST.
- data - dane do żądania
- params - parametry żądania

```
BASE_URL = "http://127.0.0.1:5000"
HEADERS = {
    "Authorization": "Karol Zelazowski"
}

def endpoint_test(endpoint, method="GET", data=None, params=None):
    url = f"{BASE_URL}{endpoint}"
    if method == "GET":
        response = requests.get(url, headers=HEADERS, params=params)
    else:
        response = requests.post(url, headers=HEADERS, json=data)
    print(f"Endpoint: {method} {endpoint}")
    print(f"Status code: {response.status_code}")
    print("JSON:")
    print(json.dumps(response.json(), indent=4, ensure_ascii=False))
```

Rysunek 6: Ogólna budowa żądania

2.2. Żądanie teacher-list

```
#Zapytanie o listę nauczycieli
endpoint_test("/teacher-list")
```

Rysunek 7: Zapytanie o listę nauczycieli

```
Endpoint: GET /teacher-list
Status code: 200
JSON:
[
  {
    "id_nauczyciela": 1,
    "imie": "Jan",
    "nazwisko": "Kowalski",
    "poadzone_przedmioty": "[\"matematyka\"]"
  },
  {
    "id_nauczyciela": 2,
    "imie": "Anna",
    "nazwisko": "Nowak",
    "poadzone_przedmioty": "[\"fizyka\", \"chemia\"]"
  },
  {
    "id_nauczyciela": 3,
    "imie": "Marek",
    "nazwisko": "Wiśniewski",
    "poadzone_przedmioty": "[\"historia\"]"
  },
  {
    "id_nauczyciela": 4,
    "imie": "Ewa",
    "nazwisko": "Zielińska",
    "poadzone_przedmioty": "[\"biologia\"]"
  },
  {
    "id_nauczyciela": 5,
    "imie": "Tomasz",
    "nazwisko": "Krawczyk",
    "poadzone_przedmioty": "[\"matematyka\", \"fizyka\"]"
  }
]
```

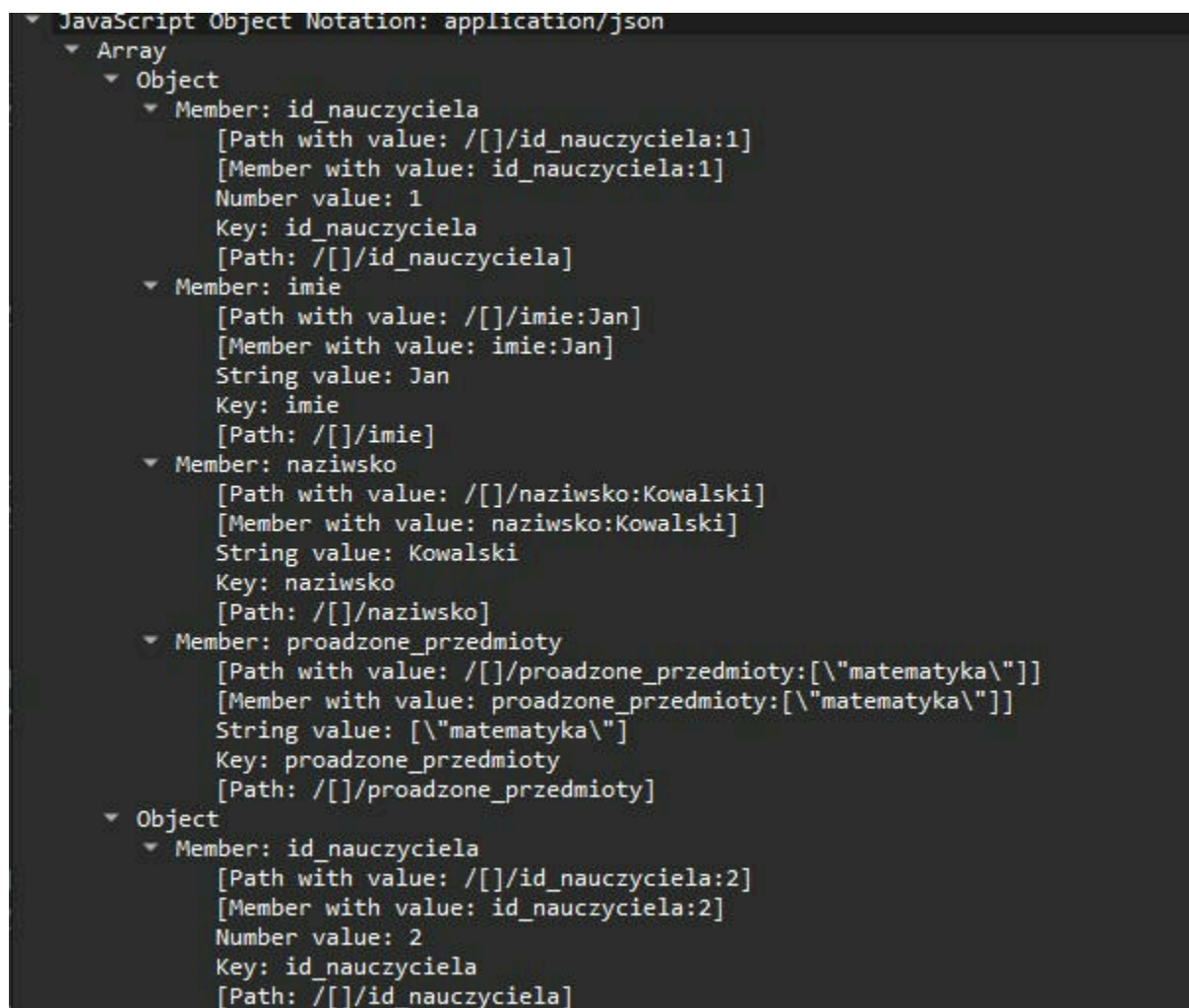
Rysunek 8: Odpowiedź w terminalu

```
▼ Hypertext Transfer Protocol
  ▼ GET /teacher-list HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): GET /teacher-list HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /teacher-list
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/teacher-list]
      [HTTP request 1/1]
      [Response in frame: 2229]
```

Rysunek 9: Żądanie przechwycone w Wireshark'u

```
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 200 OK\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 12:55:16 GMT\r\n
      Content-Type: application/json\r\n
      ▼ Content-Length: 675\r\n
        [Content length: 675]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.615316000 seconds]
      [Request in frame: 2224]
      [Request URI: http://127.0.0.1:5000/teacher-list]
      File Data: 675 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Array
      ▶ Object
      ▶ Object
      ▶ Object
      ▶ Object
      ▶ Object
```

Rysunek 10: Odpowiedź przechwycona w Wireshark'u cz. 1



Rysunek 11: Odpowiedź przechwycona w Wireshark'u cz. 2

2.3. Zapytanie teacher-details

2.3.1. Test z poprawnym parametrem

```

# test pozytywny
endpoint_test("/teacher-details", params={"id_nauczyciela": 1})

```

Rysunek 12: Żądanie o dane szczegółowe konkretnego nauczyciela

```

Endpoint: GET /teacher-details
Status code: 200
JSON:
{
  "dostepnosc": [
    {
      "dostepny_do": "17:00:00",
      "dostepny_od": "09:00:00"
    }
  ],
  "email": "jan.kowalski@example.com",
  "id_nauczyciela": 1,
  "id_nauczyciela": 1,
  "id_nauczyciela": 1,
  "imie": "Jan",
  "nazwisko": "Kowalski",
  "numer_telefonu": "123456789",
  "ocena_nauczyciela": 4.5,
  "opis": "Doświadczony nauczyciel matematyki",
  "prowadzone_przedmioty": "[\"matematyka\"]",
  "stawka": 100,
  "waluta": "PLN"
}

```

Rysunek 13: Odpowiedź w terminalu

```

▶ Frame 2993: 254 bytes on wire (2032 bits), 254 bytes captured (2032 bits) on interface
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 59482, Dst Port: 5000, Seq: 1, Ack: 1, Len:
▼ Hypertext Transfer Protocol
  ▼ GET /teacher-details?id_nauczyciela=1 HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): GET /teacher-details?id_nauczyciela=1 HTTP/1.1\r\n]
    Request Method: GET
    ▼ Request URI: /teacher-details?id_nauczyciela=1
      Request URI Path: /teacher-details
      ▼ Request URI Query: id_nauczyciela=1
        Request URI Query Parameter: id_nauczyciela=1
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/teacher-details?id_nauczyciela=1]
      [HTTP request 1/1]
      [Response in frame: 2997]

```

Rysunek 14: Żądanie przechwycone w Wireshark'u

```
▶ Frame 2997: 449 bytes on wire (3592 bits), 449 bytes captured (3592 bits) on interface  
▶ Null/Loopback  
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 59482, Seq: 167, Ack: 211, L  
▶ [2 Reassembled TCP Segments (571 bytes): #2995(166), #2997(405)]  
▼ Hypertext Transfer Protocol  
  HTTP/1.1 200 OK\r\n  
    [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]  
    Response Version: HTTP/1.1  
    Status Code: 200  
    [Status Code Description: OK]  
    Response Phrase: OK  
    Server: Werkzeug/3.1.3 Python/3.11.2\r\n  
    Date: Tue, 17 Dec 2024 13:00:27 GMT\r\n  
    Content-Type: application/json\r\n  
    Content-Length: 405\r\n    [Content length: 405]  
    Connection: close\r\n  
    \r\n  
    [HTTP response 1/1]  
    [Time since request: 0.503041000 seconds]  
    [Request in frame: 2993]  
    [Request URI: http://127.0.0.1:5000/teacher-details?id_nauczyciela=1]  
    File Data: 405 bytes  
▼ JavaScript Object Notation: application/json  
  Object  
    Member: dostepnosc  
    Member: email  
    Member: id_nauczyciela  
    Member: imie  
    Member: nazwisko  
    Member: numer_telefonu  
    Member: ocena_nauczyciela  
    Member: opis  
    Member: prowadzone_przedmioty  
    Member: stawka  
    Member: waluta
```

Rysunek 15: Odpowiedź na żądanie przechwycona w Wireshark'u

2.3.2. Test z złym parametrem

```
# test negatywny  
endpoint_test("/teacher-details", params={"id_nauczyciela": 88})
```

Rysunek 16: Żądanie ze złym parametrem

```
Endpoint: GET /teacher-details  
Status code: 404  
JSON:  
{  
  "error": "Nauczyciel nie istnieje"  
}
```

Rysunek 17: Odpowiedź w terminalu


```
▶ Frame 3369: 255 bytes on wire (2040 bits), 255 bytes captured (2040 bits) on interface Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 59547, Dst Port: 5000, Seq: 1, Ack: 1, Len:
▼ Hypertext Transfer Protocol
  ▼ GET /teacher-details?id_nauczyciela=88 HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): GET /teacher-details?id_nauczyciela=88 HTTP/1.1\r\n
      Request Method: GET
    ▼ Request URI: /teacher-details?id_nauczyciela=88
      Request URI Path: /teacher-details
      ▼ Request URI Query: id_nauczyciela=88
        Request URI Query Parameter: id_nauczyciela=88
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/teacher-details?id_nauczyciela=88]
      [HTTP request 1/1]
      [Response in frame: 3373]
```

Rysunek 18: Żądanie przechwycone w Wireshark'u

```
▶ Frame 3373: 85 bytes on wire (680 bits), 85 bytes captured (680 bits) on interface \D
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 59547, Seq: 173, Ack: 212, L
▶ [2 Reassembled TCP Segments (213 bytes): #3371(172), #3373(41)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 404 NOT FOUND\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 404 NOT FOUND\r\n
      Response Version: HTTP/1.1
      Status Code: 404
      [Status Code Description: Not Found]
      Response Phrase: NOT FOUND
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 13:02:58 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 41\r\n
      [Content length: 41]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.389845000 seconds]
      [Request in frame: 3369]
      [Request URI: http://127.0.0.1:5000/teacher-details?id_nauczyciela=88]
      File Data: 41 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Object
      ▼ Member: error
        [Path with value: /error:Nauczyciel nie istnieje]
        [Member with value: error:Nauczyciel nie istnieje]
        String value: Nauczyciel nie istnieje
        Key: error
        [Path: /error]
```

Rysunek 19: Odpowiedź na żądanie przechwycona w Wireshark'u

2.4. Zapytanie book-lessons

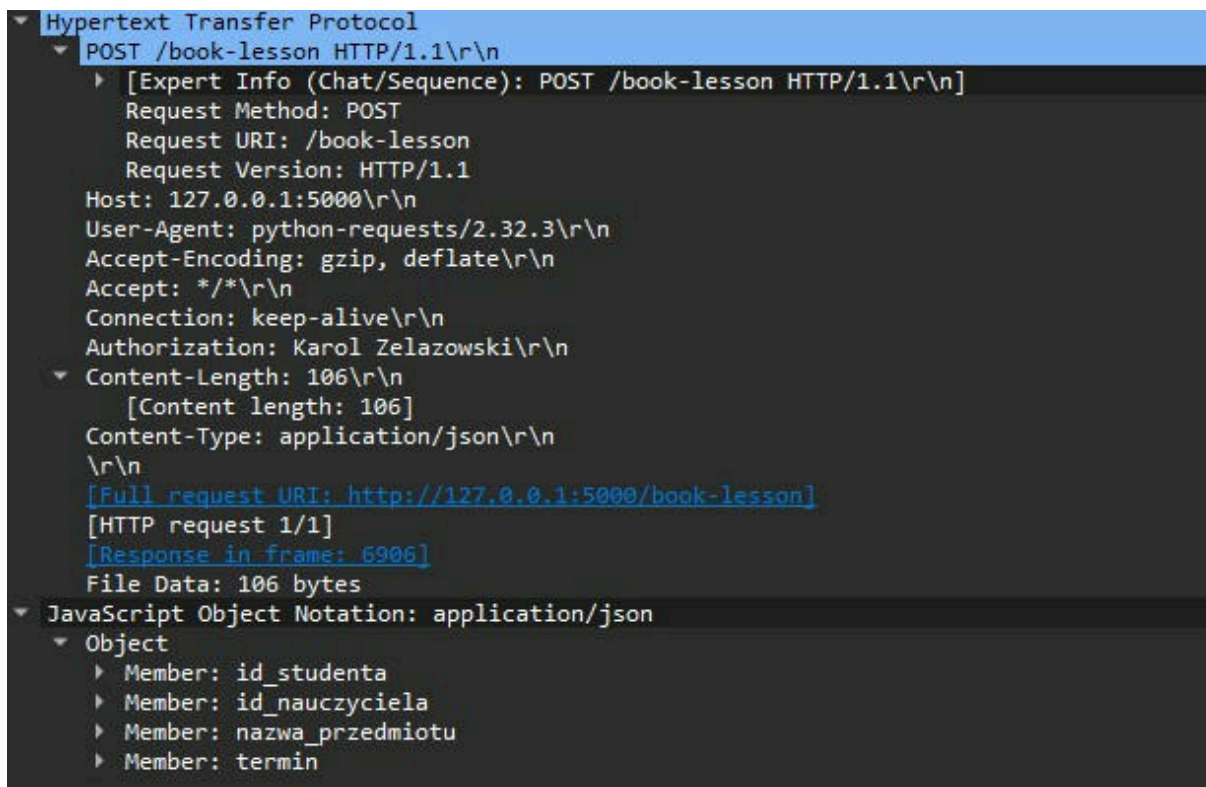
2.4.1. Test z poprawnym terminem

```
lekcja_do_zarezerwowania = {  
    "id_studenta": 1,  
    "id_nauczyciela": 1,  
    "nazwa_przedmiotu": "matematyka",  
    "termin": "2024-12-17T10:00:00"  
}  
# test pozytywny  
endpoint_test("/book-lesson", method="POST", data=lekcja_do_zarezerwowania)
```

Rysunek 20: Żądanie z poprawnymi danymi

```
Endpoint: POST /book-lesson  
Status code: 200  
JSON:  
{  
    "message": "Lekcja została zarezerwowana"  
}
```

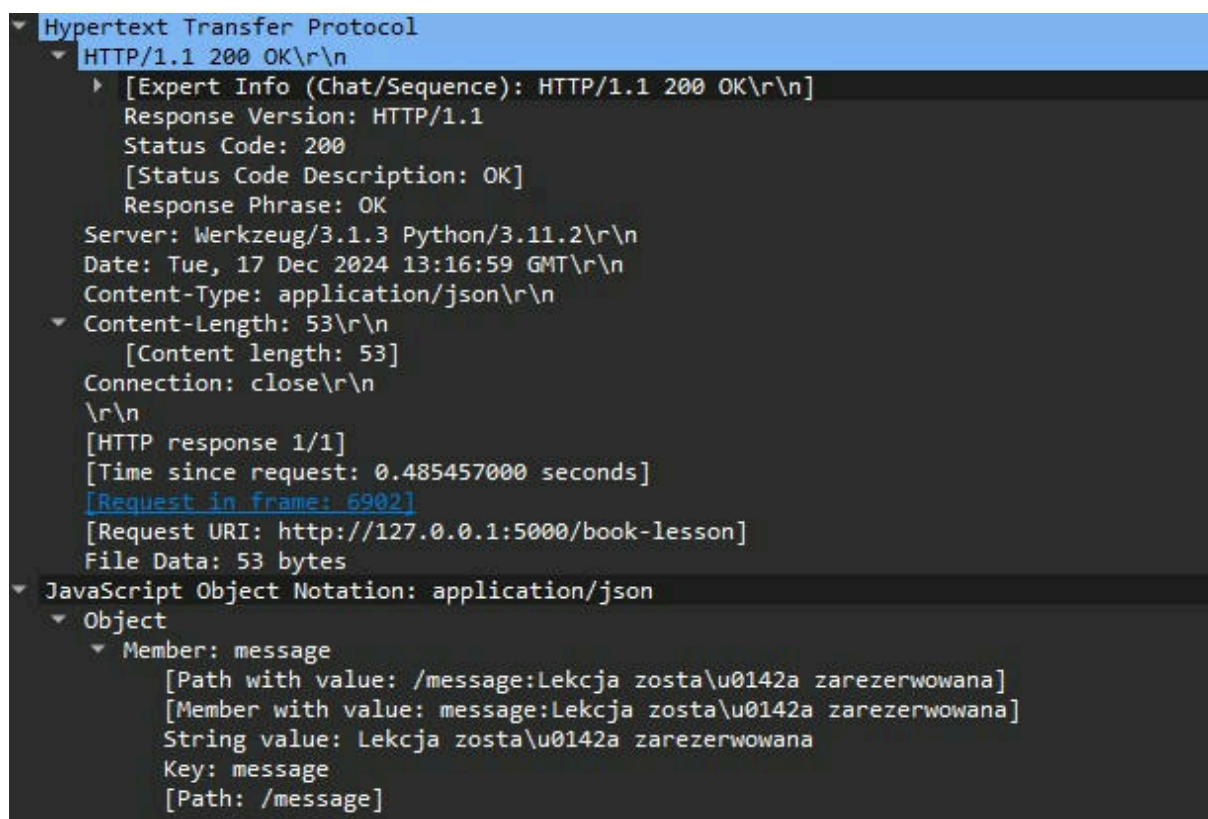
Rysunek 21: Odpowiedź w terminalu



The image shows a Wireshark packet capture of an HTTP POST request. The packet list on the left shows a POST request to /book-lesson. The packet details pane on the right shows the request structure, including the request line, headers, and the JSON body. The JSON body contains the lesson details: id_studenta, id_nauczyciela, nazwa_przedmiotu, and termin.

```
Hypertext Transfer Protocol  
  POST /book-lesson HTTP/1.1\r\n  
    [Expert Info (Chat/Sequence): POST /book-lesson HTTP/1.1\r\n      Request Method: POST  
      Request URI: /book-lesson  
      Request Version: HTTP/1.1  
      Host: 127.0.0.1:5000\r\n      User-Agent: python-requests/2.32.3\r\n      Accept-Encoding: gzip, deflate\r\n      Accept: */*\r\n      Connection: keep-alive\r\n      Authorization: Karol Zelazowski\r\n      Content-Length: 106\r\n      [Content length: 106]  
      Content-Type: application/json\r\n      \r\n      [Full request URI: http://127.0.0.1:5000/book-lesson]  
      [HTTP request 1/1]  
      [Response in frame: 6906]  
      File Data: 106 bytes  
  JavaScript Object Notation: application/json  
    Object  
      Member: id_studenta  
      Member: id_nauczyciela  
      Member: nazwa_przedmiotu  
      Member: termin
```

Rysunek 22: Żądanie przechwycone w Wireshark'u



Rysunek 23: Odpowiedź na żądanie przechwycona w Wireshark'u

2.4.2. Test z zajętym terminem

```
lekcja_juz_zajeta = {
    "id_studenta": 1,
    "id_nauczyciela": 1,
    "termin": "2024-12-17T10:00:00",
    "nazwa_przedmiotu": "Matematyka"
}
# test negatywny
endpoint_test("/book-lesson", method="POST", data=lekcja_juz_zajeta)
```

Rysunek 24: Żądanie z zajętym terminem

```
Endpoint: POST /book-lesson
Status code: 400
JSON:
{
  "error": "Nauczyciel jest już zajęty w podanym terminie!"
}
```

Rysunek 25: Odpowiedź w terminalu


```

▶ Frame 7659: 150 bytes on wire (1200 bits), 150 bytes captured (1200 bits) on interface
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 60148, Dst Port: 5000, Seq: 244, Ack: 1, Len
▶ [2 Reassembled TCP Segments (349 bytes): #7657(243), #7659(106)]
▼ Hypertext Transfer Protocol
  ▼ POST /book-lesson HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): POST /book-lesson HTTP/1.1\r\n]
      Request Method: POST
      Request URI: /book-lesson
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
    ▼ Content-Length: 106\r\n
      [Content length: 106]
      Content-Type: application/json\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/book-lesson]
      [HTTP request 1/1]
      [Response in frame: 7663]
      File Data: 106 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Object
      ▶ Member: id_studenta
      ▶ Member: id_nauczyciela
      ▶ Member: termin
      ▶ Member: nazwa_przedmiotu

```

Rysunek 26: Żądanie przechwycone w Wireshark'u

```
▶ Frame 7663: 118 bytes on wire (944 bits), 118 bytes captured (944 bits) on interface
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 60148, Seq: 175, Ack: 350, L
▶ [2 Reassembled TCP Segments (248 bytes): #7661(174), #7663(74)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 400 BAD REQUEST\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 400 BAD REQUEST\r\n]
      Response Version: HTTP/1.1
      Status Code: 400
      [Status Code Description: Bad Request]
      Response Phrase: BAD REQUEST
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 13:19:29 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 74\r\n
      [Content length: 74]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.003573000 seconds]
      [Request in frame: 7659]
      [Request URI: http://127.0.0.1:5000/book-lesson]
      File Data: 74 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Object
      ▼ Member: error
        [Path with value: /error:Nauczyciel jest ju\u0017c zaj\u00119ty w podanym termi
        [Member with value: error:Nauczyciel jest ju\u0017c zaj\u00119ty w podanym term
        String value: Nauczyciel jest ju\u0017c zaj\u00119ty w podanym terminie!
        Key: error
        [Path: /error]
```

Rysunek 27: Odpowiedź na żądanie przechwycona w Wireshark'u

2.4.3. Test ze złym terminem

```
lekcja_niepoprawny Termin = {
    "id_studenta": 1,
    "id_nauczyciela": 1,
    "termin": "2024-12-19T22:00:00",
    "nazwa_przedmiotu": "matematyka"
}

# test negatywny
endpoint_test("/book-lesson", method="POST", data=lekcja_niepoprawny_termin)
```

Rysunek 28: Żądanie ze złym terminem

```
Endpoint: POST /book-lesson
Status code: 400
JSON:
{
    "error": "Nauczyciel nie jest dost\u0119pny w tych godzinach"
}
```

Rysunek 29: Odpowiedź w terminalu


```
▶ Frame 8304: 150 bytes on wire (1200 bits), 150 bytes captured (1200 bits) on interface  
▶ Null/Loopback  
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1  
▶ Transmission Control Protocol, Src Port: 60217, Dst Port: 5000, Seq: 244, Ack: 1, Len  
▶ [2 Reassembled TCP Segments (349 bytes): #8302(243), #8304(106)]  
▼ Hypertext Transfer Protocol  
  ▼ POST /book-lesson HTTP/1.1\r\n  
    ▶ [Expert Info (Chat/Sequence): POST /book-lesson HTTP/1.1\r\n      Request Method: POST  
      Request URI: /book-lesson  
      Request Version: HTTP/1.1  
      Host: 127.0.0.1:5000\r\n      User-Agent: python-requests/2.32.3\r\n      Accept-Encoding: gzip, deflate\r\n      Accept: */*\r\n      Connection: keep-alive\r\n      Authorization: Karol Zelazowski\r\n    ▼ Content-Length: 106\r\n      [Content length: 106]  
      Content-Type: application/json\r\n      \r\n      [Full request URI: http://127.0.0.1:5000/book-lesson]  
      [HTTP request 1/1]  
      [Response in frame: 8308]  
      File Data: 106 bytes  
▼ JavaScript Object Notation: application/json  
  ▼ Object  
    ▶ Member: id_studenta  
    ▶ Member: id_nauczyciela  
    ▶ Member: termin  
    ▶ Member: nazwa_przedmiotu
```

Rysunek 30: Żądanie przechwycone w Wireshark'u

```
▶ Frame 8308: 112 bytes on wire (896 bits), 112 bytes captured (896 bits) on interface
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 60217, Seq: 175, Ack: 350, L
▶ [2 Reassembled TCP Segments (242 bytes): #8306(174), #8308(68)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 400 BAD REQUEST\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 400 BAD REQUEST\r\n]
      Response Version: HTTP/1.1
      Status Code: 400
      [Status Code Description: Bad Request]
      Response Phrase: BAD REQUEST
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 13:22:10 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 68\r\n
      [Content length: 68]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.020342000 seconds]
      [Request in frame: 8304]
      [Request URI: http://127.0.0.1:5000/book-lesson]
      File Data: 68 bytes
    ▼ JavaScript Object Notation: application/json
      ▼ Object
        ▼ Member: error
          [Path with value: /error:Nauczyciel nie jest dost\u0119pny w tych godzinach]
          [Member with value: error:Nauczyciel nie jest dost\u0119pny w tych godzinach]
          String value: Nauczyciel nie jest dost\u0119pny w tych godzinach
          Key: error
          [Path: /error]
```

Rysunek 31: Odpowiedź na żądanie przechwycona w Wireshark'u

2.5. Zapytanie add-teacher

2.5.1. Test z poprawnymi danymi

```
poprawny_nauczyciel = {
    "imie": "Stefan",
    "nazwisko": "Banach",
    "prowadzone_przedmioty": ["matematyka"],
    "opis": "Wybitny matematyk.",
    "ocena_nauczyciela": 4.9,
    "numer_telefonu": "987654221",
    "stawka": 150,
    "waluta": "PLN",
    "email": "stf.banach@gmail.com"
}
# test pozytywny
endpoint_test("/add-teacher", method="POST", data=poprawny_nauczyciel)
```

Rysunek 32: Żądanie z poprawnymi danymi

```
Endpoint: POST /add-teacher
Status code: 200
JSON:
{
  "id_nauczyciela": 6
}
```

Rysunek 33: Odpowiedź w terminalu

Frame 22313: 281 bytes on wire (2248 bits), 281 bytes captured (2248 bits) on interface Null/Loopback

Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1

Transmission Control Protocol, Src Port: 51883, Dst Port: 5000, Seq: 244, Ack: 1, Len: [2 Reassembled TCP Segments (480 bytes): #22311(243), #22313(237)]

Hypertext Transfer Protocol

- POST /add-teacher HTTP/1.1\r\n
 - [Expert Info (Chat/Sequence): POST /add-teacher HTTP/1.1\r\n]
 - Request Method: POST
 - Request URI: /add-teacher
 - Request Version: HTTP/1.1
 - Host: 127.0.0.1:5000\r\n
 - User-Agent: python-requests/2.32.3\r\n
 - Accept-Encoding: gzip, deflate\r\n
 - Accept: */*\r\n
 - Connection: keep-alive\r\n
 - Authorization: Karol Zelazowski\r\n
 - Content-Length: 237\r\n
 - [Content length: 237]
 - Content-Type: application/json\r\n\r\n
 - [Full request URI: http://127.0.0.1:5000/add-teacher]
 - [HTTP request 1/1]
 - [Response in frame: 22317]
 - File Data: 237 bytes
- JavaScript Object Notation: application/json
 - Object
 - Member: imie
 - Member: nazwisko
 - Member: prowadzone_przedmioty
 - Member: opis
 - Member: ocena_nauczyciela
 - Member: numer_telefonu
 - Member: stawka
 - Member: waluta
 - Member: email

Rysunek 34: Żądanie przechwycone w Wireshark'u


```
▶ Frame 22317: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 51883, Seq: 166, Ack: 481, L
▶ [2 Reassembled TCP Segments (191 bytes): #22315(165), #22317(26)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 200 OK\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 14:19:47 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 26\r\n
      [Content length: 26]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.021417000 seconds]
      [Request in frame: 22313]
      [Request URI: http://127.0.0.1:5000/add-teacher]
      File Data: 26 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Object
      ▼ Member: id_nauczyciela
        [Path with value: /id_nauczyciela:6]
        [Member with value: id_nauczyciela:6]
        Number value: 6
        Key: id_nauczyciela
        [Path: /id_nauczyciela]
```

Rysunek 35: Odpowiedź na żądanie przechwycona w Wireshark'u

2.5.2. Test ze złymi danymi

```
nauczyciel_niepoprawny = {
    "imie": "Juliusz",
    "nazwisko": "Mickiewicz",
    "prowadzone_przedmioty": "fizyka",
    "opis": "Ziutek",
    "ocena_nauczyciela": 1.0,
    "numer_telefonu": "123123123",
    "stawka": 150,
    "waluta": "PLN",
}

# test negatywny
endpoint_test("/add-teacher", method="POST", data=nauczyciel_niepoprawny)
```

Rysunek 36: Żądanie z brakującymi danymi

```
Endpoint: POST /add-teacher
Status code: 400
JSON:
{
  "error": "Brak wymaganej wartości: 'email'"
}
```

Rysunek 37: Odpowiedź w terminalu

The image shows a Wireshark packet capture of an HTTP POST request and its response. The packet list on the left shows Frame 22730 (231 bytes on wire) and Frame 22734 (187 bytes on wire). The packet details pane shows the following information:

- Frame 22730: 231 bytes on wire (1848 bits), 231 bytes captured (1848 bits) on interface Null/Loopback
- Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
- Transmission Control Protocol, Src Port: 51954, Dst Port: 5000, Seq: 244, Ack: 1, Len: 430
- [2 Reassembled TCP Segments (430 bytes): #22728(243), #22730(187)]
- Hypertext Transfer Protocol
 - POST /add-teacher HTTP/1.1\r\n
 - [Expert Info (Chat/Sequence): POST /add-teacher HTTP/1.1\r\n]
 - Request Method: POST
 - Request URI: /add-teacher
 - Request Version: HTTP/1.1
 - Host: 127.0.0.1:5000\r\n
 - User-Agent: python-requests/2.32.3\r\n
 - Accept-Encoding: gzip, deflate\r\n
 - Accept: */*\r\n
 - Connection: keep-alive\r\n
 - Authorization: Karol Zelazowski\r\n
 - Content-Length: 187\r\n
 - [Content length: 187]
 - Content-Type: application/json\r\n\r\n
 - [Full request URI: http://127.0.0.1:5000/add-teacher]
 - [HTTP request 1/1]
 - [Response in frame: 22734]
 - File Data: 187 bytes
- JavaScript Object Notation: application/json
 - Object
 - Member: imie
 - Member: nazwisko
 - Member: prowadzone_przedmioty
 - Member: opis
 - Member: ocena_nauczyciela
 - Member: numer_telefonu
 - Member: stawka
 - Member: waluta

Rysunek 38: Żądanie przechwycone w Wireshark'u

```
▶ Frame 22734: 99 bytes on wire (792 bits), 99 bytes captured (792 bits) on interface \
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 51954, Seq: 175, Ack: 431, L
▶ [2 Reassembled TCP Segments (229 bytes): #22732(174), #22734(55)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 400 BAD REQUEST\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 400 BAD REQUEST\r\n]
      Response Version: HTTP/1.1
      Status Code: 400
      [Status Code Description: Bad Request]
      Response Phrase: BAD REQUEST
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 14:21:55 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 55\r\n
      [Content length: 55]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.001435000 seconds]
      [Request in frame: 22730]
      [Request URI: http://127.0.0.1:5000/add-teacher]
      File Data: 55 bytes
    ▼ JavaScript Object Notation: application/json
      ▼ Object
        ▼ Member: error
          [Path with value: /error:Brak wymaganej warto\u015bci: 'email']
          [Member with value: error:Brak wymaganej warto\u015bci: 'email']
          String value: Brak wymaganej warto\u015bci: 'email'
          Key: error
          [Path: /error]
```

Rysunek 39: Odpowiedź na żądanie przechwycona w Wireshark'u

2.6. Zapytanie get-lessons

2.6.1. Test z poprawnymi parametrami

```
# test pozytywny
endpoint_test("/get-lessons", params={"id_studenta": 1, "data_od": "2024-12-01", "data_do": "2024-12-20"})
```

Rysunek 40: Żądanie z poprawnymi parametrami


```
Endpoint: GET /get-lessons
Status code: 200
JSON:
[
  {
    "data_lekcji": "Tue, 10 Dec 2024 00:00:00 GMT",
    "id_nauczyciela": 1,
    "imie_nauczyciela": "Jan",
    "nazwisko_nauczyciela": "Kowalski",
    "przedmiot": "matematyka"
  },
  {
    "data_lekcji": "Wed, 11 Dec 2024 00:00:00 GMT",
    "id_nauczyciela": 1,
    "imie_nauczyciela": "Jan",
    "nazwisko_nauczyciela": "Kowalski",
    "przedmiot": "matematyka"
  },
  {
    "data_lekcji": "Thu, 12 Dec 2024 00:00:00 GMT",
    "id_nauczyciela": 1,
    "imie_nauczyciela": "Jan",
    "nazwisko_nauczyciela": "Kowalski",
    "przedmiot": "matematyka"
  },
  {
    "data_lekcji": "Wed, 11 Dec 2024 00:00:00 GMT",
    "id_nauczyciela": 1,
    "imie_nauczyciela": "Jan",
    "nazwisko_nauczyciela": "Kowalski",
    "przedmiot": "matematyka"
  },
  {
    "data_lekcji": "Tue, 17 Dec 2024 00:00:00 GMT",
    "id_nauczyciela": 1,
    "imie_nauczyciela": "Jan",
    "nazwisko_nauczyciela": "Kowalski",
    "przedmiot": "matematyka"
  }
]
```

Rysunek 41: Odpowiedź w terminalu

```
▶ Frame 23655: 285 bytes on wire (2280 bits), 285 bytes captured (2280 bits) on interfa
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 52086, Dst Port: 5000, Seq: 1, Ack: 1, Len:
▼ Hypertext Transfer Protocol
  ▼ GET /get-lessons?id_studenta=1&data_od=2024-12-01&data_do=2024-12-20 HTTP/1.1\r\n
    ▶ [Expert Info (Chat/Sequence): GET /get-lessons?id_studenta=1&data_od=2024-12-01
      Request Method: GET
    ▼ Request URI: /get-lessons?id_studenta=1&data_od=2024-12-01&data_do=2024-12-20
      Request URI Path: /get-lessons
      ▼ Request URI Query: id_studenta=1&data_od=2024-12-01&data_do=2024-12-20
        Request URI Query Parameter: id_studenta=1
        Request URI Query Parameter: data_od=2024-12-01
        Request URI Query Parameter: data_do=2024-12-20
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/get-lessons?id_studenta=1&data_od=2024-12
      [HTTP request 1/1]
      [Response in frame: 23659]
```

Rysunek 42: Żądanie przechwycone w Wireshark'u


```
▶ Frame 23659: 982 bytes on wire (7856 bits), 982 bytes captured (7856 bits) on inter
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 52086, Seq: 167, Ack: 242,
▶ [2 Reassembled TCP Segments (1104 bytes): #23657(166), #23659(938)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 200 OK\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 14:25:09 GMT\r\n
      Content-Type: application/json\r\n
      Content-Length: 938\r\n
      [Content length: 938]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.019361000 seconds]
      [Request in frame: 23655]
      Request URI: http://127.0.0.1:5000/get-lessons?id_studenta=1&data_od=2024-12-01
      File Data: 938 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Array
      ▼ Object
        ▼ Member: data_lekcji
          [Path with value: /[]/data_lekcji:Tue, 10 Dec 2024 00:00:00 GMT]
          [Member with value: data_lekcji:Tue, 10 Dec 2024 00:00:00 GMT]
          String value: Tue, 10 Dec 2024 00:00:00 GMT
          Key: data_lekcji
          [Path: /[]/data_lekcji]
        ▼ Member: id_nauczyciela
          [Path with value: /[]/id_nauczyciela:1]
          [Member with value: id_nauczyciela:1]
          Number value: 1
```

Rysunek 43: Odpowiedź na żądanie z przechwyconą w Wireshark'u

2.6.2. Test ze złym parametrem

```
# test negatywny
endpoint_test(["/get-lessons", params={"id_studenta": 16, "data_pocatkowa": "2024-12-01", "data_koncowa": "2024-12-17"}])
```

Rysunek 44: Żądanie ze złym parametrem

```
Endpoint: GET /get-lessons
Status code: 404
JSON:
{
  "error": "Student nie istnieje"
}
```

Rysunek 45: Odpowiedź w terminalu

```

▶ Frame 24113: 299 bytes on wire (2392 bits), 299 bytes captured (2392 bits) on interfa
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 52154, Dst Port: 5000, Seq: 1, Ack: 1, Len:
▼ Hypertext Transfer Protocol
  ▼ GET /get-lessons?id_studenta=16&data_początkowa=2024-12-01&data_koncowa=2024-12-17
    ▶ [Expert Info (Chat/Sequence): GET /get-lessons?id_studenta=16&data_początkowa=20
      Request Method: GET
    ▼ Request URI: /get-lessons?id_studenta=16&data_początkowa=2024-12-01&data_koncow
      Request URI Path: /get-lessons
      ▼ Request URI Query: id_studenta=16&data_początkowa=2024-12-01&data_koncowa=20
        Request URI Query Parameter: id_studenta=16
        Request URI Query Parameter: data_początkowa=2024-12-01
        Request URI Query Parameter: data_koncowa=2024-12-17
      Request Version: HTTP/1.1
      Host: 127.0.0.1:5000\r\n
      User-Agent: python-requests/2.32.3\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept: */*\r\n
      Connection: keep-alive\r\n
      Authorization: Karol Zelazowski\r\n
      \r\n
      [Full request URI: http://127.0.0.1:5000/get-lessons?id_studenta=16&data_początkow
      [HTTP request 1/1]
      [Response in frame: 24117]

```

Rysunek 46: Żądanie przechwycone w Wireshark'u

```

▶ Frame 24117: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface \
▶ Null/Loopback
▶ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▶ Transmission Control Protocol, Src Port: 5000, Dst Port: 52154, Seq: 173, Ack: 256, L
▶ [2 Reassembled TCP Segments (210 bytes): #24115(172), #24117(38)]
▼ Hypertext Transfer Protocol
  ▼ HTTP/1.1 404 NOT FOUND\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 404 NOT FOUND\r\n]
      Response Version: HTTP/1.1
      Status Code: 404
      [Status Code Description: Not Found]
      Response Phrase: NOT FOUND
      Server: Werkzeug/3.1.3 Python/3.11.2\r\n
      Date: Tue, 17 Dec 2024 14:27:22 GMT\r\n
      Content-Type: application/json\r\n
    ▼ Content-Length: 38\r\n
      [Content length: 38]
      Connection: close\r\n
      \r\n
      [HTTP response 1/1]
      [Time since request: 0.013995000 seconds]
      [Request in frame: 24113]
      [Request URI: http://127.0.0.1:5000/get-lessons?id_studenta=16&data_poczatkowa=202
      File Data: 38 bytes
  ▼ JavaScript Object Notation: application/json
    ▼ Object
      ▼ Member: error
        [Path with value: /error:Student nie istnieje]
        [Member with value: error:Student nie istnieje]
        String value: Student nie istnieje
        Key: error
        [Path: /error]

```

Rysunek 47: Odpowiedź na żądanie przechwycona w Wireshark'u