**Міністерство освіти і науки України**

**Національний технічний університет України «Київський політехнічний**

**інститут імені Ігоря Сікорського"**

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

З лабораторної роботи № 2 з дисципліни

«Безпека програмного забезпечення»

“**Авторизаційний протокол OAuth2**”

| **Виконав(ла)** | *ІП-13 Бабіч Денис* |  |  |
| --- | --- | --- | --- |
|  | (шифр, прізвище, ім'я, по батькові) |  |  |

| **Перевірив(ла)** | *Соколовський В. В.* |  |  |
| --- | --- | --- | --- |
|  | (шифр, прізвище, ім'я, по батькові) |  |  |

Київ 2024

**ЛАБОРАТОРНА РОБОТА № 2**

**Тема роботи**: Авторизаційний протокол OAuth2.

**Мета роботи**: Засвоїти базові навички OAuth2 авторизаційного протокола.

**Основне завдання:**

1. Використовуючи наведені налаштування, створити запит на отримання токену через client\_credential grant.

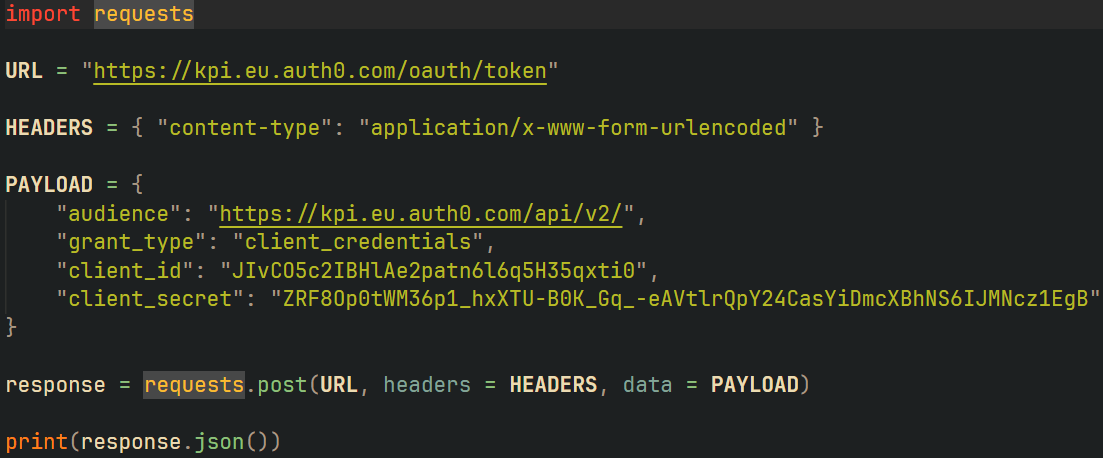


Рисунок 1.1 – Код для звернення до ендпоінту з метою отримання токену

У цьому коді реалізується запит на отримання токена доступу за допомогою протоколу OAuth 2.0. Далі задається змінна URL, що містить адрес веб-сервісу, до якого надсилається запит, у даному випадку це URL для отримання токена в Auth0. Наступна змінна HEADERS визначає заголовки запиту, де вказується, що тип контенту - "application/x-www-form-urlencoded", що є стандартним форматом для передачі даних у формі ключ-значення. Потім створюється словник PAYLOAD, який містить дані, що будуть надіслані в тілі запиту і цей словник включає тип запиту, ідентифікатор клієнта та секрет клієнта. Далі виконується POST-запит за допомогою методу requests.post, який приймає URL, заголовки та дані, результат запиту зберігається в змінній response.

У результаті виконання коду на рисунку 1.1 було отримано наступну відповідь, яка містить доступний токен авторизації типу Bearer, що дозволяє доступ до API, такий токен складається з трьох частин, шифрується та підписується, що забезпечує безпеку під час передачі даних. Домен дії токена включає права на читання та створення користувачів, що вказує на можливість взаємодії з користувацькими даними

{

'access\_token': 'eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IjVCZTlBZFhrMERaUjhmR1dZYjdkViJ9.eyJpc3MiOiJodHRwczovL2twaS5ldS5hdXRoMC5jb20vIiwic3ViIjoiSkl2Q081YzJJQkhsQWUycGF0bjZsNnE1SDM1cXh0aTBAY2xpZW50cyIsImF1ZCI6Imh0dHBzOi8va3BpLmV1LmF1dGgwLmNvbS9hcGkvdjIvIiwiaWF0IjoxNzI3NjUwNzA3LCJleHAiOjE3Mjc3MzcxMDcsInNjb3BlIjoicmVhZDp1c2VycyBjcmVhdGU6dXNlcnMiLCJndHkiOiJjbGllbnQtY3JlZGVudGlhbHMiLCJhenAiOiJKSXZDTzVjMklCSGxBZTJwYXRuNmw2cTVIMzVxeHRpMCJ9.fXnVM6clYkUp3HsYBPnvzsOgBWxP6rejao7depzsPu5wlQApwtnxuTsptevvY-YFfJjS9imIS9JQpV\_T317nKSo9HGrpeZQy2E\_2xiMl4ht1SLvjXQD9z-nDMUvaw0uz93uYgwCGCr5U1reBkaxhMmNLCjeOHXueyCMZUnsaVuuZg32S\_apaRAOPgmx5ump4-4aBtu4Axk-o2WBxFEs6aPzMtOwu7zF3wCTZ5juHzH-rukKBrKXCl7CJmOX-q5JQjOe0DC5ZUwu9PfThsYju6Fer4vymJqdL7Swyz6chaC-83DdR90pecSh4lY2RLB5z2B12JOr0P0U9U\_-KUl1QQg',

'scope': 'read:users create:users',

'expires\_in': 86400,

'token\_type': 'Bearer'

}

Тип токена "Bearer" вказує, що для доступу до ресурсу необхідно надіслати токен доступу в заголовку Authorization у форматі "Bearer {token}". Це означає, що будь-хто, хто має цей токен, може отримати доступ до захищених ресурсів, що робить важливим правильне управління та захист токенів від несанкціонованого використання.

1. Створити юзера з власним email в системі, використовуючи метод https://auth0.com/docs/api/management/v2#!/Users/post\_users

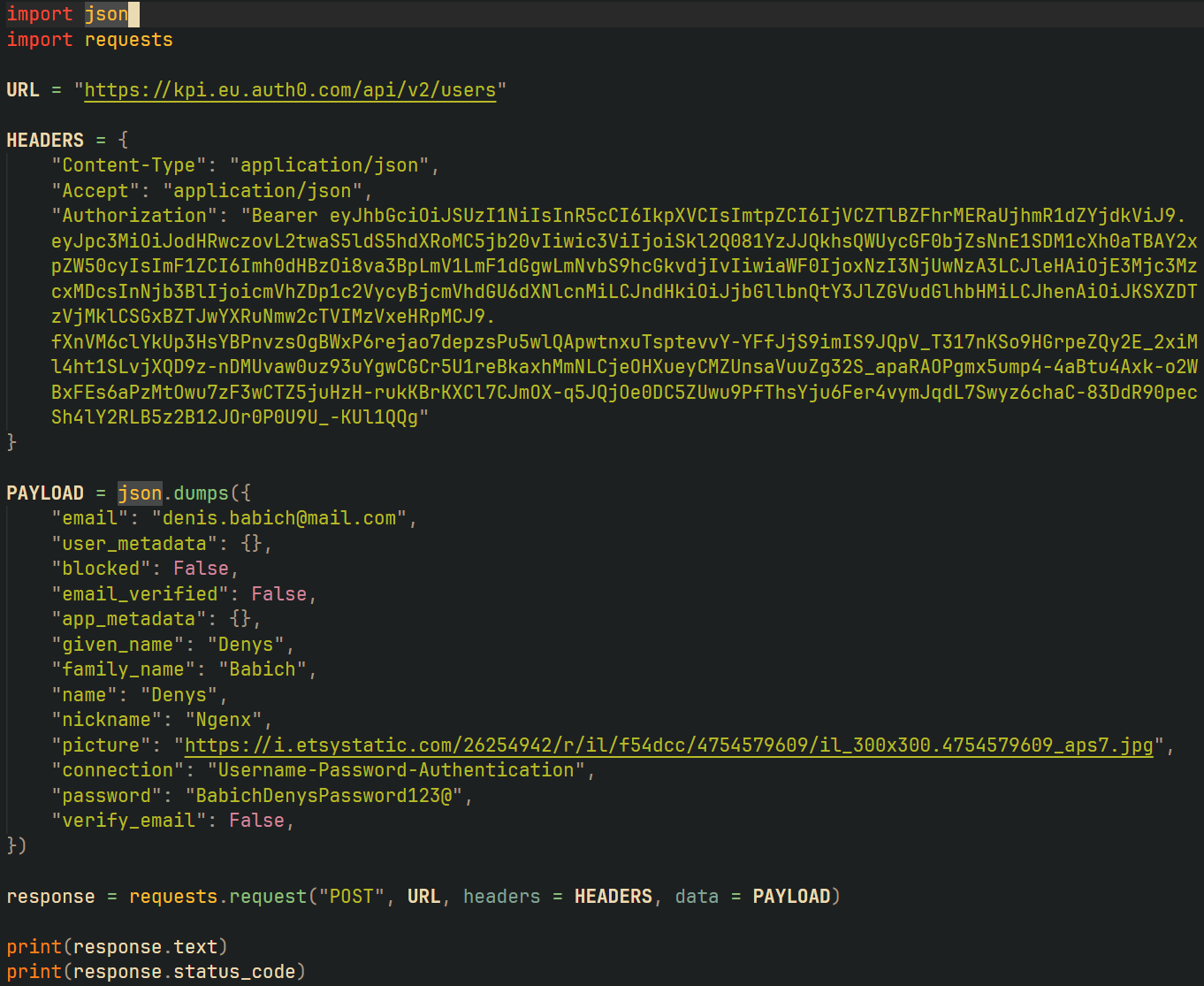


Рисунок 1.2 – Код для створення запиту на створення користувача

Цей код виконує HTTP-запит для створення нового користувача через API Auth0, використовуючи метод POST. Спочатку задається URL-адресу API, заголовки, що включають тип вмісту, прийняті формати та авторизаційний токен. Далі код формується корисне навантаження у форматі JSON, яке містить електронну адресу, метадані користувача, підтвердження електронної адреси, дані про користувача, такі як ім'я, прізвище, нікнейм, посилання на зображення профілю, метод підключення та пароль. Після формування корисного навантаження код виконує POST-запит до API з зазначеними заголовками та даними.

У результаті виконання коду на рисунку 1.2 було повернуто статус-код 201, що свідчить про те, що користувача було успішно створено.

{"blocked":false,"created\_at":"2024-09-29T23:06:03.895Z","email":"denis.babich@mail.com","email\_verified":false,"family\_name":"Babich","given\_name":"Denys","identities":[{"connection":"Username-Password-Authentication","user\_id":"66f9dd5b99d33f07238e0a45","provider":"auth0","isSocial":false}],"name":"Denys","nickname":"Ngenx","picture":"[https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il\_300x300.4754579609\_aps7.jpg","updated\_at":"2024-09-29T23:06:03.896Z","user\_id":"auth0|66f9dd5b99d33f07238e0a45","user\_metadata](https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il_300x300.4754579609_aps7.jpg%22,%22updated_at%22:%222024-09-29T23:06:03.896Z%22,%22user_id%22:%22auth0%7C66f9dd5b99d33f07238e0a45%22,%22user_metadata)":{}}

**Додаткове завдання:**

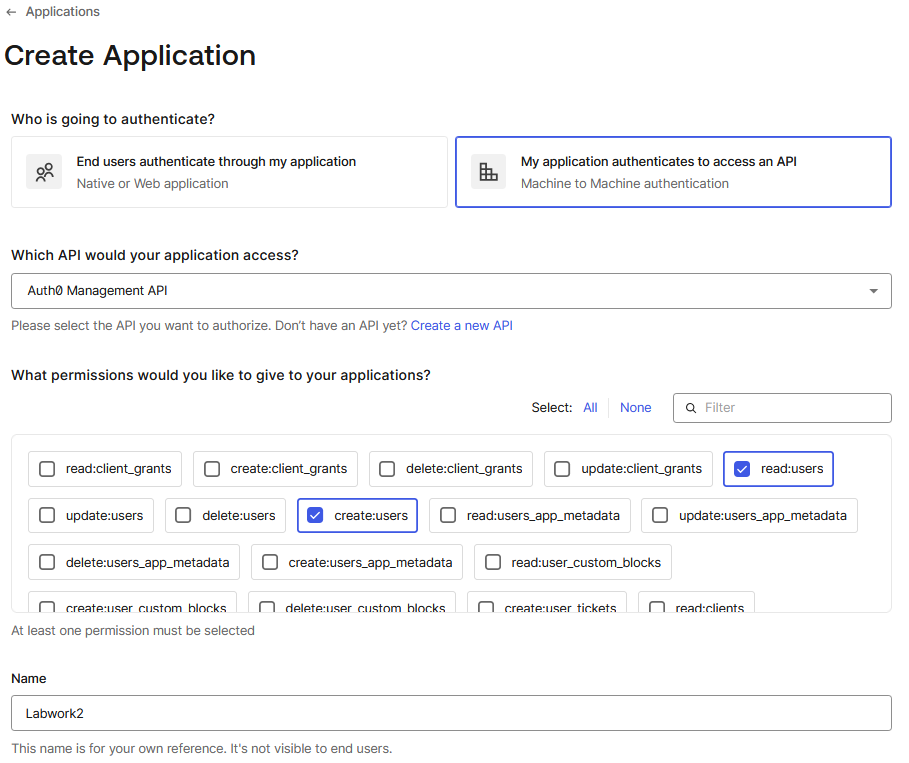


Рисунок 1.3 – Створення власного ендпоінту, де у користувача є права для створення та читання користувачів



Рисунок 1.4 – Приклад коду, для звернення до ендпоінту та отримання токену

Результат виконання коду з рисунку 1.4 можна побачити нижче:

{

"access\_token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IkxUUEVRenBxeDNkU3VYY2lyUGlHSSJ9.eyJpc3MiOiJodHRwczovL2Rldi1oZnpiNnNldXRoNWplc3lwLmV1LmF1dGgwLmNvbS8iLCJzdWIiOiJNdkdha0Q2YmRWUjBHZXpVQldnY0ViQnF0amVnUW9wR0BjbGllbnRzIiwiYXVkIjoiaHR0cHM6Ly9kZXYtaGZ6YjZzZXV0aDVqZXN5cC5ldS5hdXRoMC5jb20vYXBpL3YyLyIsImlhdCI6MTcyNzY1MjQxNCwiZXhwIjoxNzI3NzM4ODE0LCJzY29wZSI6InJlYWQ6dXNlcnMgY3JlYXRlOnVzZXJzIiwiZ3R5IjoiY2xpZW50LWNyZWRlbnRpYWxzIiwiYXpwIjoiTXZHYWtENmJkVlIwR2V6VUJXZ2NFYkJxdGplZ1FvcEcifQ.NvvIrY5H\_xFESL\_oiVYxbv0iwH\_-7PnpqxMHgawk-VpKOlAhhZC3gcknxTicUOFYIyKu\_fc\_FDFMwexgMmlZXZ1IiKUkzxpziRGPVZ5xV1U02cgTNd4vEcLHwb0iDzREpKL-O48N7M-K44\_6LQ\_gRlsNZg7i5caxlaib0RMJFwlbkmVjPw6C7roak84JswIWcVZNwJzIpbvJtlsv4xoi9FSDHOFNZ91erJIyPz6OZPZkQYLg0haUnOKE4ohTrIn\_rWfB4uea2mof94zIsk2-gKw8HOKe6N\_vD3brwIqYkok\_LZJIU12\_6DEwe1WpnxGezvYo8QSKB7ctPeHn0QgBYA",

"scope":"read:users create:users",

"expires\_in":86400,

"token\_type":"Bearer"}

import json

import http.client

from typing import Tuple

def get\_access\_token(url: str, payload: str) -> str:

connection = http.client.HTTPSConnection(url)

headers = {'content-type': "application/json"}

connection.request("POST", "/oauth/token", payload, headers)

data = connection.getresponse().read()

return json.loads(data.decode("utf-8"))['access\_token']

def create\_user(url: str, access\_token: str, payload: str) -> Tuple[str, int]:

headers = {

"Accept": "application/json",

"Content-Type": "application/json",

"Authorization": f"Bearer {access\_token}"

}

connection = http.client.HTTPSConnection(url)

connection.request("POST", "/api/v2/users", payload, headers)

result = connection.getresponse()

data = result.read()

return data.decode("utf-8"), result.status

if \_\_name\_\_ == "\_\_main\_\_":

url = "dev-hfzb6seuth5jesyp.eu.auth0.com"

payload\_token = json.dumps({

"client\_id": "MvGakD6bdVR0GezUBWgcEbBqtjegQopG",

"client\_secret": "D796h3AHQg-JJxsa2OiMmfbJ3tcW9qSowOY-FaDWQ0vxVdsGuLln1xNz45lUtWRo",

"audience": "https://dev-hfzb6seuth5jesyp.eu.auth0.com/api/v2/",

"grant\_type": "client\_credentials"

})

access\_token = get\_access\_token("dev-hfzb6seuth5jesyp.eu.auth0.com", payload\_token)

payload\_user = json.dumps({

"email": "denis.babich@test.com",

"user\_metadata": {},

"blocked": False,

"email\_verified": False,

"app\_metadata": {},

"given\_name": "Denys",

"family\_name": "Babich",

"name": "Denys",

"nickname": "TestUser",

"picture": "https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il\_300x300.4754579609\_aps7.jpg",

"connection": "Username-Password-Authentication",

"password": "BabichDenysPassword123@",

"verify\_email": False,

})

response\_data, status\_code = create\_user(url, access\_token, payload\_user)

print(f"Response Data: {response\_data}")

print(f"Status Code: {status\_code}")

Вищезазначений код реалізує отримання токена доступу та створення нового користувача через API Auth0. Функція get\_access\_token виконує POST-запит до точки /oauth/token, передаючи необхідні дані для отримання токена доступу, який використовується для авторизації наступних запитів. Після цього функція create\_user здійснює POST-запит до /api/v2/users, передаючи дані нового користувача, включаючи електронну адресу, ім'я, прізвище, пароль та інші метадані.

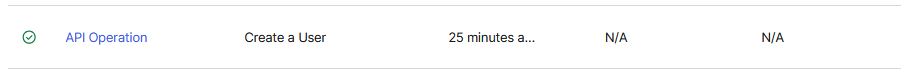


Рисунок 1.5 – Підтвердження успішної операції створення користувача у логах

У результаті запуску отримується такий вивід, який має статус-код 201, що свідчить про успішно-створеного користувача:

Response Data: {"blocked":false,"created\_at":"2024-09-29T23:42:30.424Z","email":"denis.babich@test.com","email\_verified":false,"family\_name":"Babich","given\_name":"Denys","identities":[{"connection":"Username-Password-Authentication","user\_id":"66f9e5e61d6c705c7c47b630","provider":"auth0","isSocial":false}],"name":"Denys","nickname":"TestUser","picture":"https://i.etsystatic.com/26254942/r/il/f54dcc/4754579609/il\_300x300.4754579609\_aps7.jpg","updated\_at":"2024-09-29T23:42:30.424Z","user\_id":"auth0|66f9e5e61d6c705c7c47b630","user\_metadata":{}}

Status Code: 201

**Висновок:** У ході виконання лабораторної роботи було досліджено авторизаційний протокол OAuth2, зокрема механізм client\_credentials grant. Під час роботи були проаналізовані ключові аспекти реалізації даного протоколу, які включають отримання токена доступу для авторизації запитів до API. Використовуючи API Auth0, було успішно реалізовано запит на отримання токена типу Bearer, що надає можливість взаємодії з захищеними ресурсами.

Крім того, у рамках лабораторної роботи було виконано створення нового користувача через API. Запит на створення користувача було здійснено з використанням отриманого токена, що підтвердило успішність операції через статус-код 201, який свідчить про те, що механізм авторизації працює коректно, дозволяючи створювати користувачів та виконувати інші операції.

У результаті аналізу були виявлені переваги використання OAuth2, зокрема можливість обмеження доступу до ресурсів без необхідності надання паролів користувачів. Проте, як і в будь-якій системі авторизації, залишаються потенційні ризики, пов’язані з управлінням токенами, які можуть бути скомпрометовані, якщо не дотримуватися належних заходів безпеки, таких як використання HTTPS, регулярна ротація токенів та контроль за їх використанням. Загалом, протокол OAuth2 є ефективним рішенням для організації авторизації, але потребує уважного підходу до забезпечення безпеки даних.