**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 № 3 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 № 3**

**Тема роботи**: Авторизаційний протокол OAuth2.

**Мета роботи**: Засвоєння базових навичок OAuth2 авторизаційного протокола.

**Основне завдання:**

1. **Використовуючи наведені налаштування з лабораторної роботи № 2 зробити запит на отримання user token (попередньо створеного в лабораторній роботі 2).**

Для виконання лабораторної роботи був використаний сервіс <https://auth0.com/>, у якому був створений тестовий застосунок та були визначені endpoint’и. Важливим елементом було додавання стандартного способу аутентифікації користувача, налаштування якого зображено на рисунку 1.1.

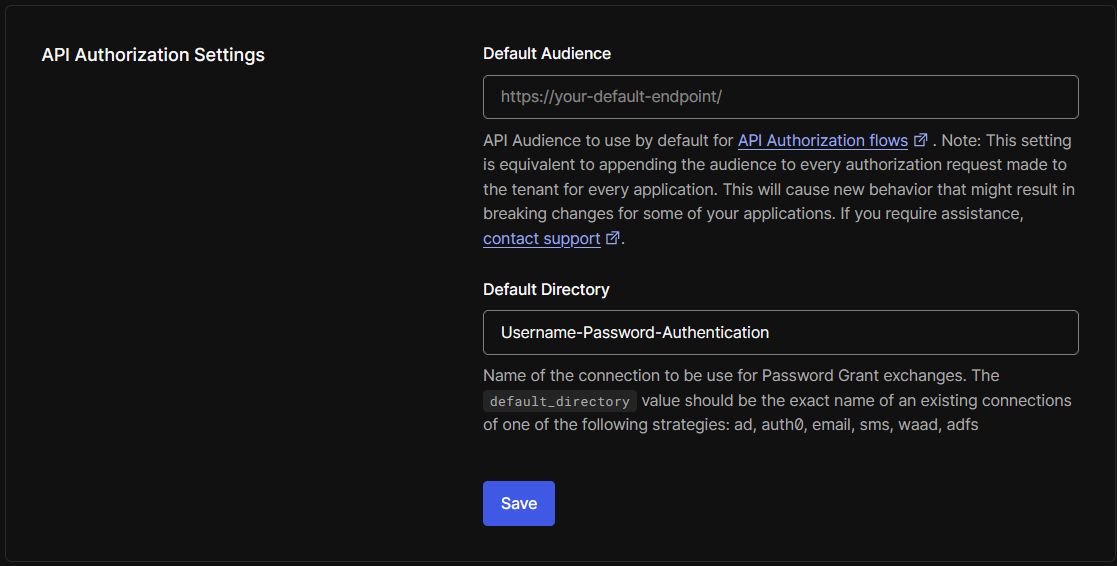


Рисунок 1.1 – Встановлення стандартного підключення за допомогою юзернейму та паролю

Для отримання токену доступу до ресурсу був використаний готовий код з лабораторної роботи № 2, який виконує POST запит до ендпоінту, отримаючи access token. Відповідний код, за допомогою якого було створено звернення можна побачити на рисунках 1.2 – 1.3.

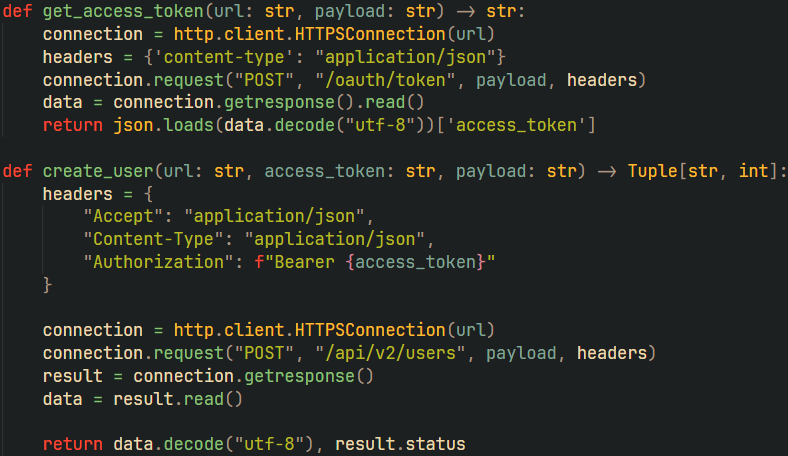


Рисунок 1.2 – Функції для створення користувача та отримання токену доступу

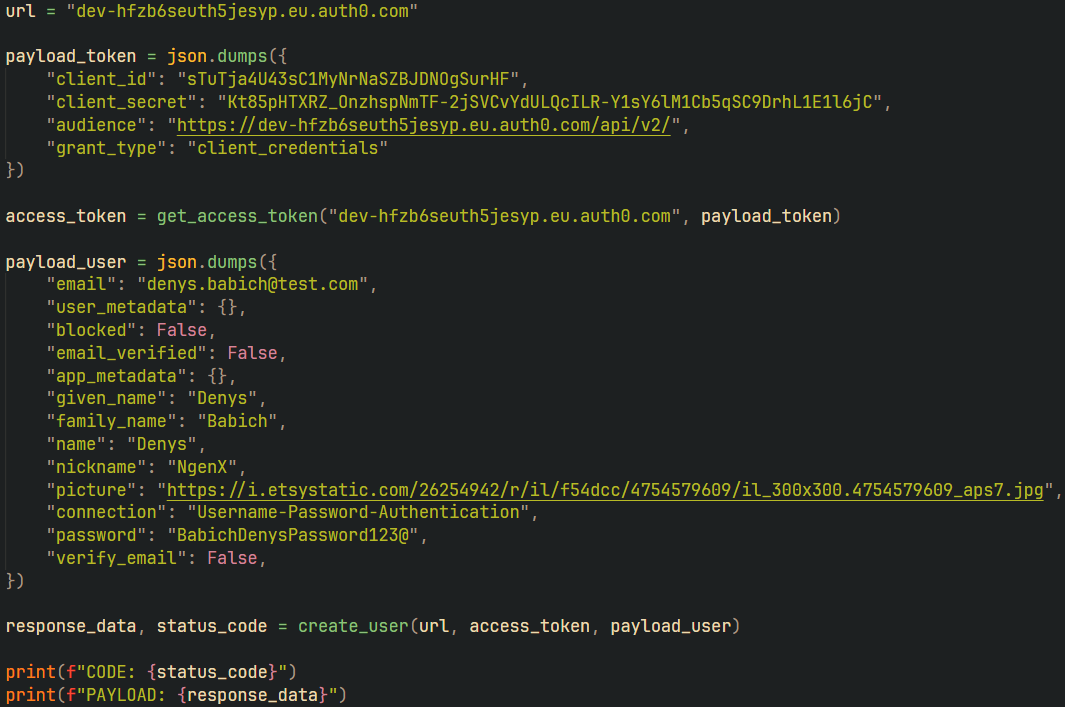


Рисунок 1.3 – Основний модуль, у якому виконуються запити

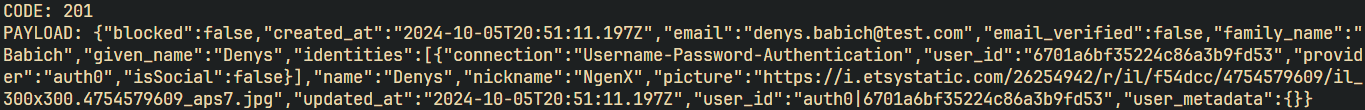


Рисунок 1.4 – Отриманий статус код 201, що означає успішне створення користувача

Для подальшого виконання роботи також потрібно додати варіанти аутентифікації до віддаленого ресурсу, тому були встановлені відповідні налаштування grant types, за допомогою яких визначається паплайн та способи обробки токенів доступу клієнта до віддаленого ресурсу. Отримані налаштування можна побачити на рисунку 1.5.

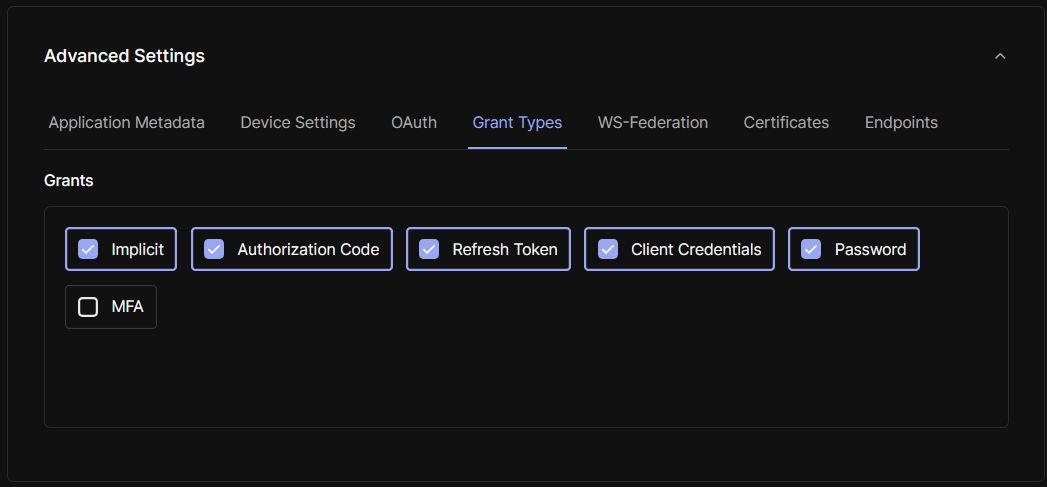


Рисунок 1.5 – Налаштування grant types для застосунку

Implicit Grant використовується у додатках, де токен доступу передається безпосередньо клієнту, без участі серверної частини. Це відбувається за сценарієм, коли користувач вводить свої облікові дані через сторінку авторизації і токен доступу (access token) повертається безпосередньо клієнту.

Authorization Code є найпоширенішим типом для серверних додатків, де токен доступу передається через код авторизації. Користувач вводить свої облікові дані на сторінці авторизації, після успішної авторизації сервер видає код авторизації і повертає його клієнту (зазвичай через редирект). Клієнт обмінює цей код на токен доступу та refresh token через сервер.

Refresh Token використовується для оновлення токену доступу після того, як час його дії закінчився, або став недійсним. Цей grant-тип зазвичай використовується разом із Authorization Code.

Client Credentials використовується для авторизації між сервісами без участі користувача, додаток сам себе авторизує за допомогою своїх облікових даних (client\_id і client\_secret), відправляючи їх на сервер авторизації.

Password передбачає, що клієнт отримує ім'я користувача і пароль шляхом реєстрації, або напряму від іншого користувача та отримує доступ до віддаленого ресурсу, шляхом авторизації через них.

Самі запити на отримання токену користувача та інші операції виконуватимуться за допомогою онлайн-сервісу Postman, який надає зручний графічний інтерфейс для створення та налаштування подібних запитів. Для отримання user-токену у тіло запиту передаються ім’я, пароль користувача, секретний заголовок запиту і доступ виконується з grant типом через пароль. Сам запит для отримання user-токену показаний на рисунку 1.6.

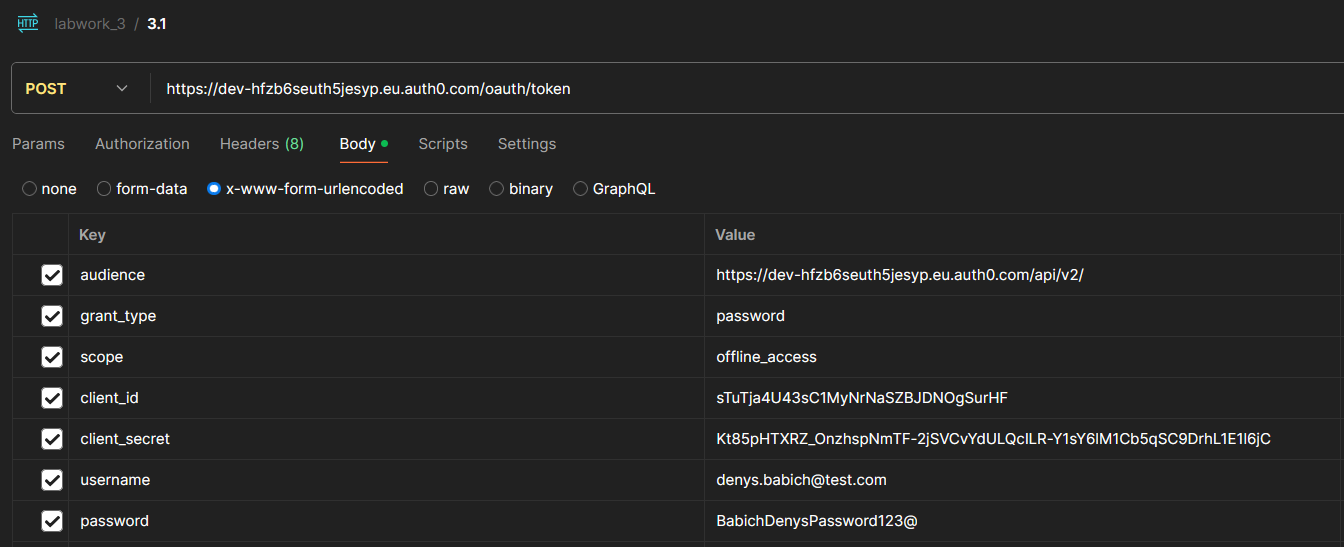


Рисунок 1.6 – Отримане тіло запиту до сервісу



Рисунок 1.7 – Отримана успішна відповідь

Отримана відповідь від сервісу включає сам токен доступу, та інші поля, зокрема scope, яке вказує на отримані права доступу, які має наданий токен доступу та refresh\_token, який використовується для отримання нового токену доступу після того, як закінчиться термін дії поточного access\_token.

{

"access\_token": "eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IkxUUEVRenBxeDNkU3VYY2lyUGlHSSJ9..ZjCHgRNzzg\_Zx7qYpyqg3C22A509\_\_w2W9xRsc2s66qT\_sfeQ-NEqxymEviyyjedlaJI83-cLNS7gzU6Q-T2\_3cobPwNatkhg6--rP\_UAhRt6kKk3Qibu84XvifHro5nX\_mf8o3W6XNaH0gu7OS3S6DVPM2QmRk4kOdZHRSM1CbQIyveZq7lufFb4x5wAB9FrOn6H-Qdze2TK4VSSLjy-eHWjyMDMZgUZTuix12agYCxOwjbZFrldFSZCODBEyabQHV3HJrHUu-xBpLak\_ceqtqv87HVeK69ix0C5K3HkWxGZ8vhJfrlRT8j9H7vg1dQSyBil6CmhXYlEgG6FPmfHA",

"refresh\_token": "rcS9idfB8nvQQBIpq8pxD6mCsq7jE8uINwbDQ-B4MVJb7",

"scope": "read:current\_user update:current\_user\_metadata delete:current\_user\_metadata create:current\_user\_metadata create:current\_user\_device\_credentials delete:current\_user\_device\_credentials update:current\_user\_identities offline\_access",

"expires\_in": 86400,

"token\_type": "Bearer"

}

1. **Отримати оновлений токен використовуючи refresh-token grant type.**

Як уже було зазначено, refresh token використовується для отримання нового токену доступу, коли закінчується термін дії старого. Для виконання такої операції можна побудувати запит, який показаний на рисунку 1.8.

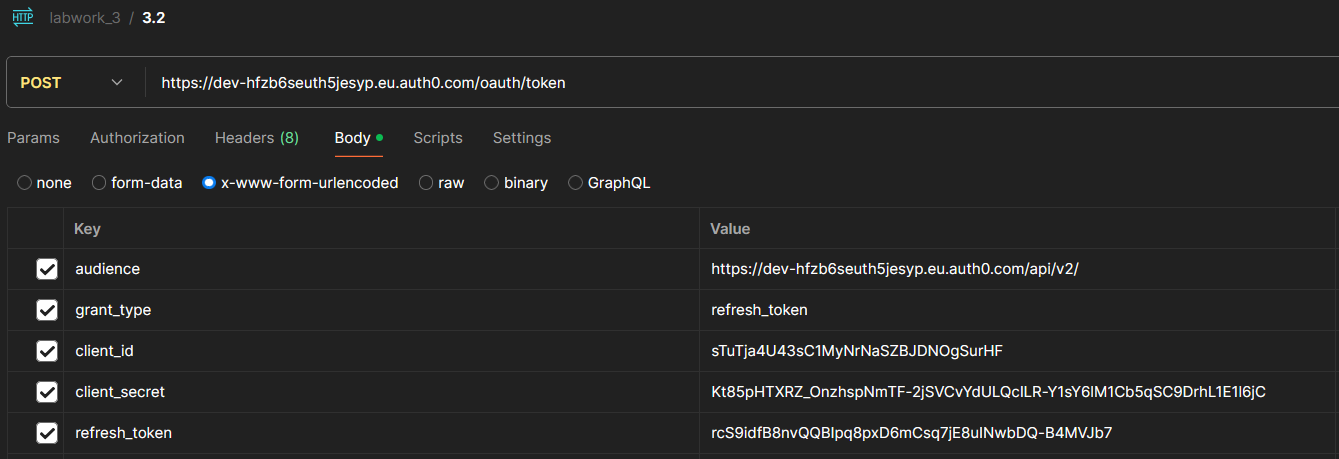


Рисунок 1.8 – Запит для отримання токену доступу за допомогою refresh-токену



Рисунок 1.9 – Отримана успішна відповідь на продовження авторизації

Тіло такої відповіді так само включає сам access-токен та область дії токену, яка вказує на отримані права доступу.

**Додаткове завдання:**

1. **Зробити запит до API для зміни пароля, токен має бути використаний з прикладу client\_credential grant прикладу.**

Для того, щоб оновити пароль буде використаний PATCH-запит, проте спочатку необхідно отримати айді клієнта, який може бути здоубтий завдяки запиту через токену доступу, який був зроблений з додаванням grant-типів openid та profile. OpenID вказує на те, що застосунок хоче виконати автентифікацію користувача (ID-токен). У свою чергу, profile означає те, що у тілі відповіді має бути така додаткова інформація, як дані користувача (ім’я, прізвище, аватар, тощо). Відповідний запит можна побачити на рисунку 1.10.

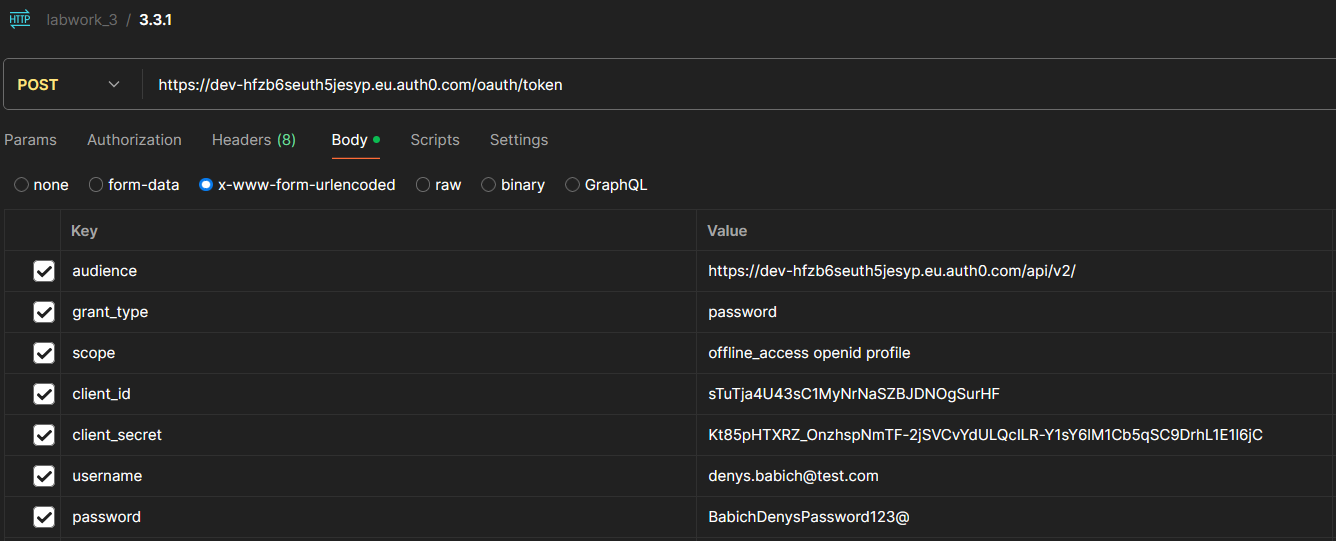


Рисунок 1.10 – Отримане тіло запиту



Рисунок 1.11 – Успішна відповідь від застосунку

Для отримання userId необхідно використати access\_token з відповіді на рисунку 1.11 та підставити у тіло запиту на endpoint з ідентифікатором /userinfo. Тіло запиту наведено на рисунку 1.12.

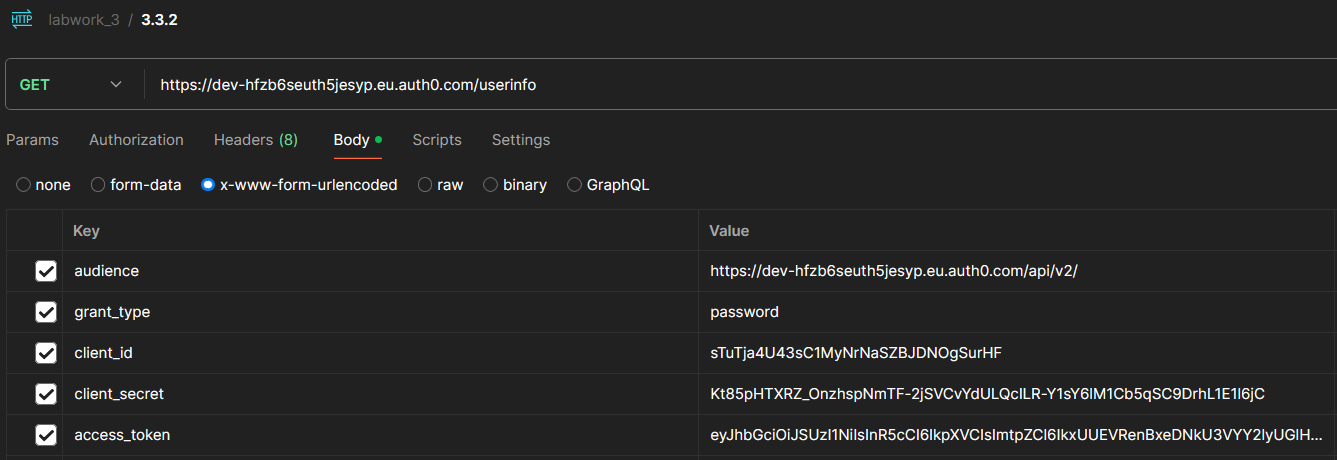


Рисунок 1.12 – Тіло запиту для отримання userId

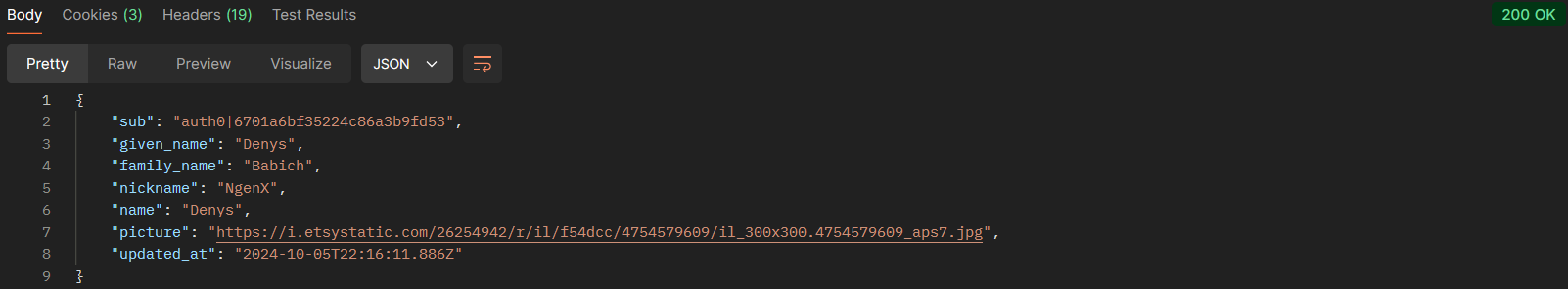


Рисунок 1.13 – Отримана успішна відповідь

{

"sub": "auth0|6701a6bf35224c86a3b9fd53",

"given\_name": "Denys",

"family\_name": "Babich",

"nickname": "NgenX",

"name": "Denys",

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

"updated\_at": "2024-10-05T22:16:11.886Z"

}

Токен доступу для зміни паролю буде виконаний не через самого користувача, а за допомогою grant-типу client\_credentials, який використовується, коли додаток прагне виконати доступ до ресурсу не від імені користувача, а від себе, зазвичай, така поведінка застосовується для комунікації напряму між клієнтським застосунком та ендпоінтом, що корисно, оскільки не вимагає залучення користувача.

Хоча є й інші альтернативні способи зміни паролю, навіть більш логічні та безпечні, проте, такий доступ через client\_credentials може застосовуватися у випадку, якщо користувач забув власний пароль й після підтвердження своєї особи альтернативними засобами (через пошту, мобільний телефон, тощо) хоче відновити доступ до свого акаунту.

Запит на отримання токену доступу можна побачити на рисунку 1.14

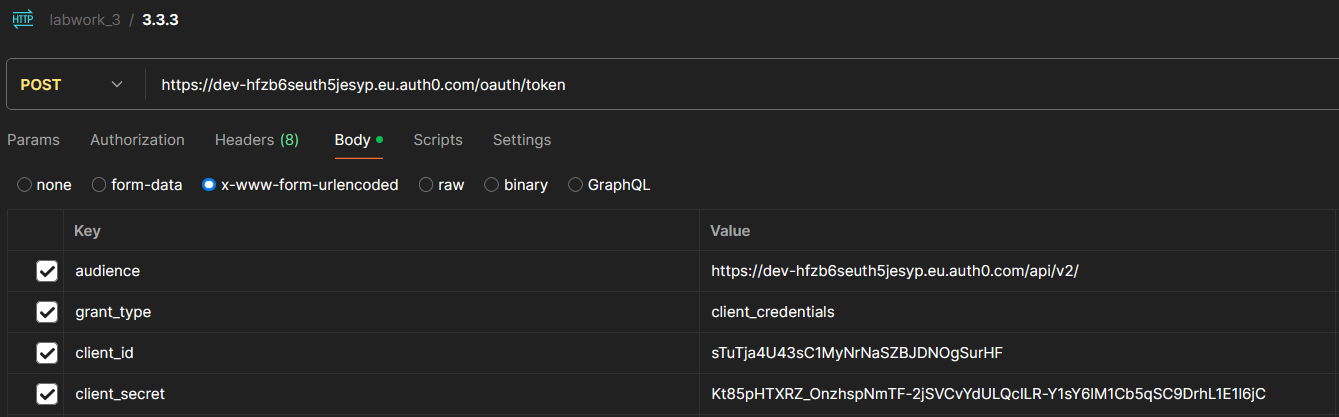


Рисунок 1.14 – Тіло запиту

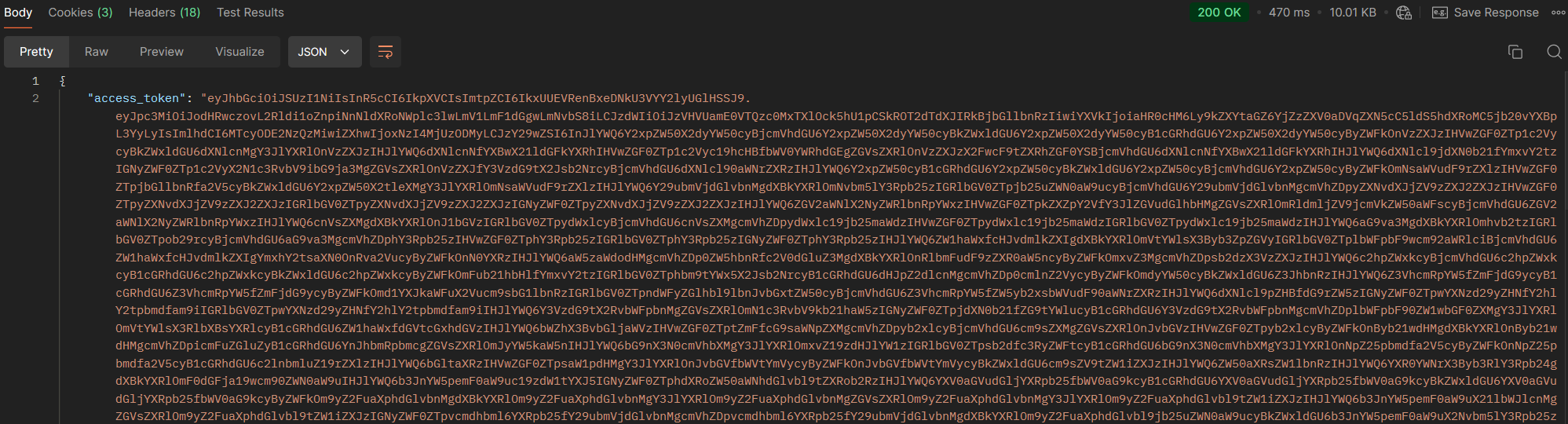


Рисунок 1.15 – Успішна відповідь від серверу

Тепер цей отриманий сесійний токен використовуватиметься для безпосередньої зміни паролю за допомогою PATCH-запиту, де у заголовок у поле Authorization передається отримане значення access-токену. Тіло з корисним навантаженням, яке зберігає поле парою наведено на рисунку 1.16.

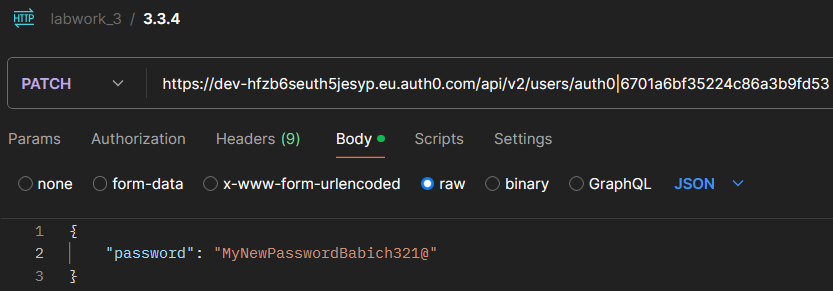


Рисунок 1.16 – Тіло запиту з новим паролем

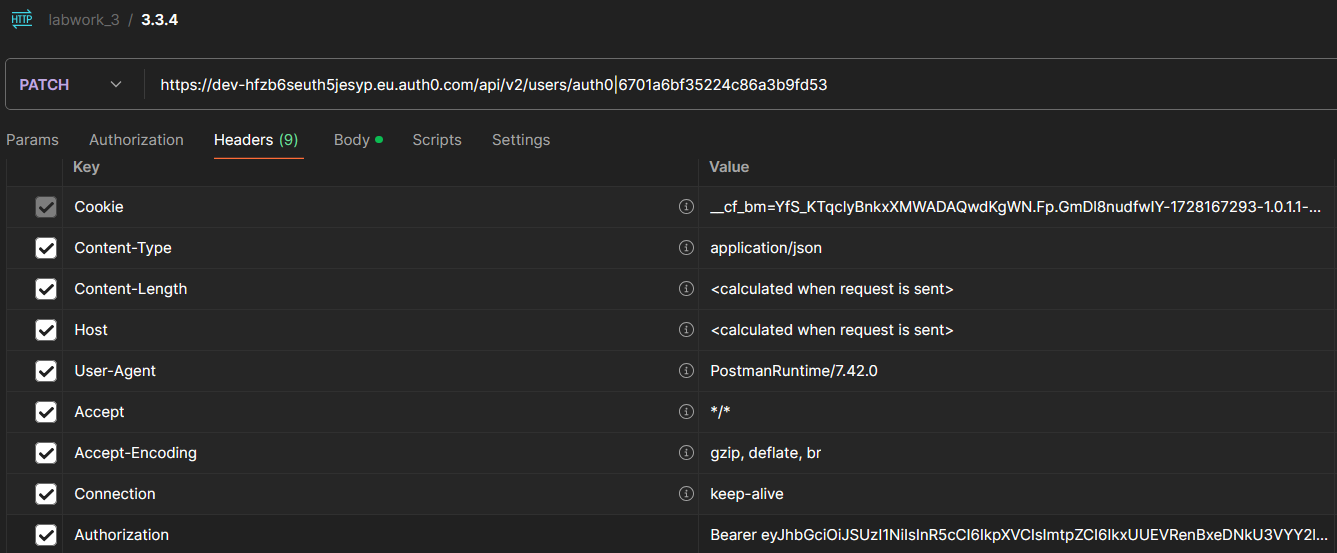


Рисунок 1.17 – Заголовок запиту, де у Authorization занесено токен доступу



Рисунок 1.18 – Отримана успішна відповідь на зміну паролю

Тепер можна виконати перевірку доступу до застосунку, спочатку буде перевірений старий пароль, тіло запиту та негативна відповідь зі статус-колом 403 наведено на рисунку 1.19.

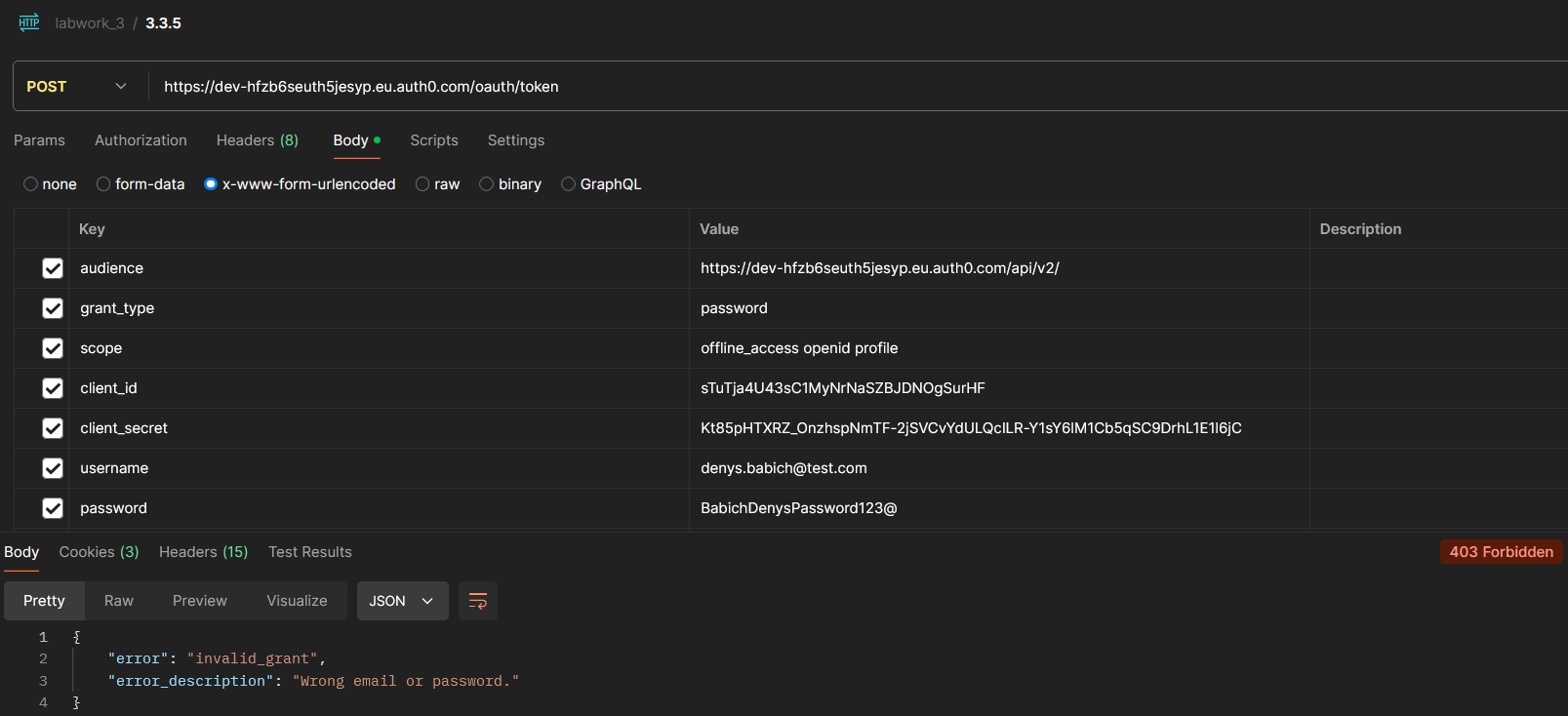


Рисунок 1.19 – Неуспішна спроба авторизації за допомогою старого пароля

У результаті стандартного запиту через grant-доступ через пароль, у тілі якого були вписані ім’я користувача, старий пароль був повернутий статус-код 403, що сигналізує про невірні облікові дані, тіло відповіді також наведено на рисунку 1.19.

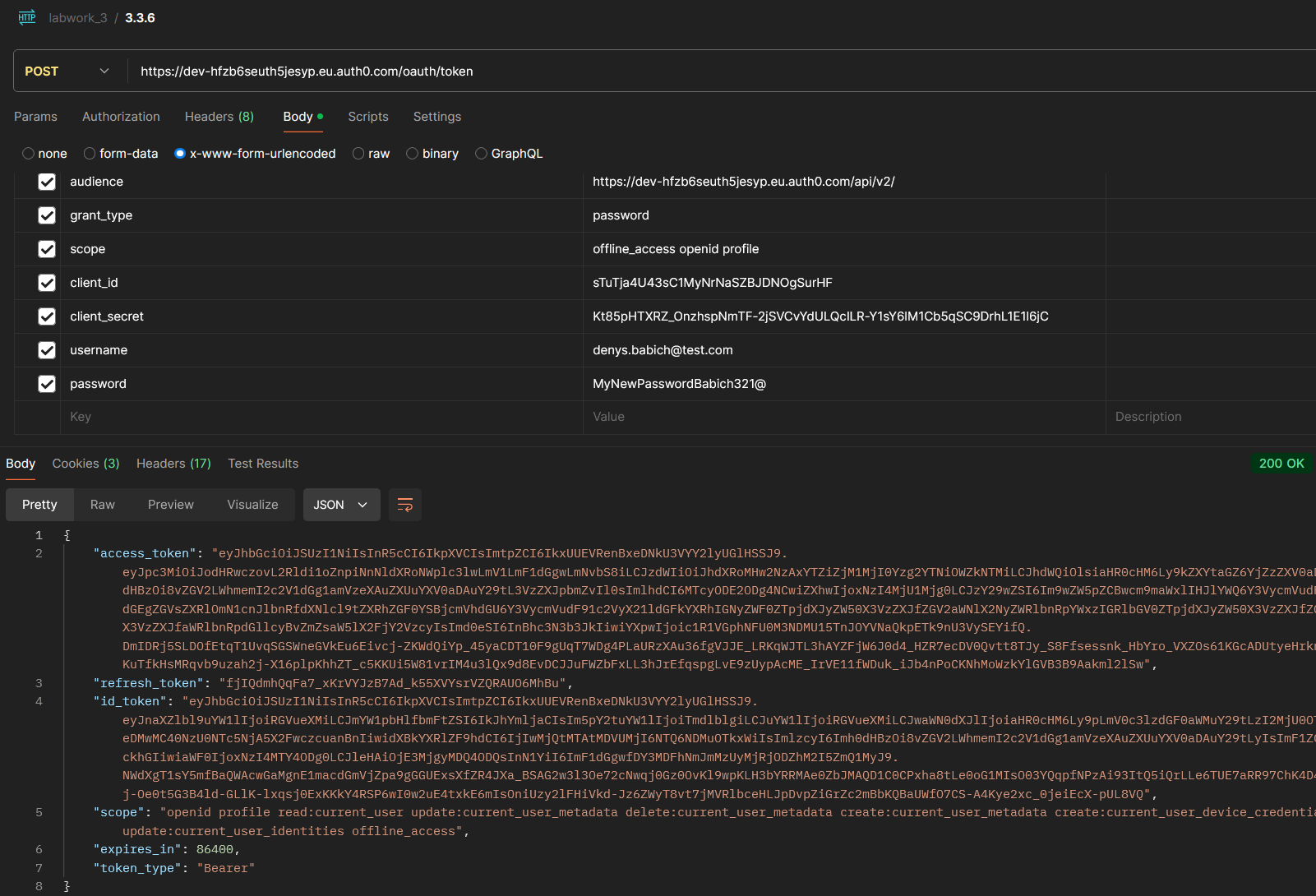


Рисунок 1.20 – Успішна авторизація за допомогою нового пароля

**Висновок:** У цій лабораторній роботі було успішно засвоєно основні навички роботи з OAuth2 авторизаційним протоколом, зокрема з використанням сервісу Auth0 для керування доступом до ресурсів. Запити та відповіді сервісу оброблялися за допомогою Postman, що дозволило вивчити структуру запитів та отриманих відповідей.

Основним завданням було отримання токену доступу (access token) та його використання для аутентифікації користувача і взаємодії з ресурсами через API. У ході роботи були розглянуті різні варіанти grant types, включаючи Password, Authorization Code, Refresh Token та Client Credentials. Особливу увагу було приділено сценаріям, коли токен доступу передається безпосередньо клієнту або ж через серверну частину. Зокрема, детально описано процес отримання нового токену через Refresh Token після закінчення терміну дії попереднього токену доступу.

Також у рамках додаткового завдання було продемонстровано застосування Client Credentials для зміни пароля користувача. У результаті роботи було виконано успішну зміну пароля користувача, підтверджену отриманими статус-кодами та відповідями сервісу, що свідчить про правильну реалізацію авторизації та керування доступом до ресурсів.