

Національний технічний університет України  
«Київський політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра Обчислювальної Техніки

Лабораторна робота №4  
з дисципліни "Безпека програмного забезпечення"  
Тема: "Засвоювання базових навичок OAuth2 авторизаційного протокола"

Виконав:

студент групи ІП-93

Домінський В.О.

Київ 2023

## **Зміст**

Завдання: .....	3
Виконання: .....	3
Висновок: .....	9
Посилання: .....	9

### Завдання:

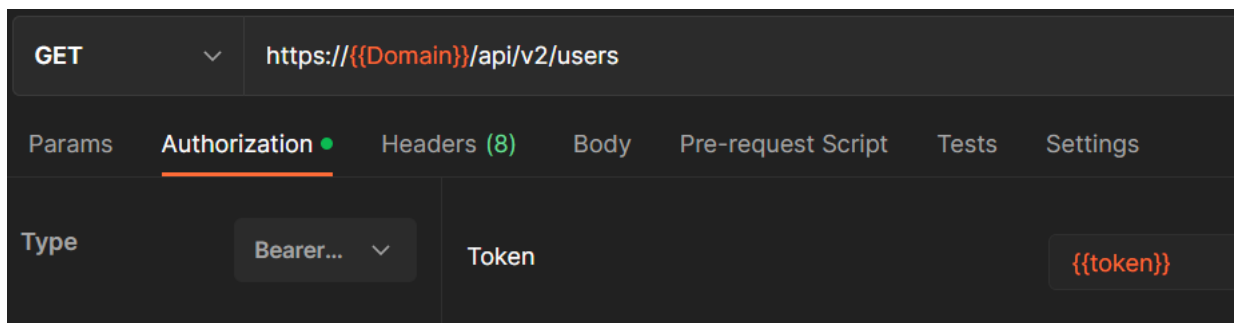
1. Використовуючи наведені налаштування з лабораторної роботи 2-3 та приведених запитів модифікувати аплікейшен, використовуючи перевірку юзера та отримання токена з auth0 (password grant type)

### Виконання:

Дана лабораторна має багато схожого з 2-ма минулими, які були виконані за допомогою Postman, тому Я буду показувати запити як ті, що написані в коді, так і ті, що були вже зроблені у 2/3 ЛР.

Для того, щоб увійти в систему, Нам треба знати логін та пароль якогось користувача. Пароль Ми можемо отримати лише від самого власника, а перше – або за допомогою запиту для отримання всіх юзерів:

### Запити:



### Відповідь:

```
"blocked": false,
"created_at": "2022-12-31T16:27:50.998Z",
"email": "susie.hoeger64@yahoo.com",
"email_verified": false,
"family_name": "Dominskyi",
"given_name": "Valentyn",
"identities": [
  {
    "user_id": "63b063065ade362990d24803",
    "provider": "auth0",
    "connection": "Username-Password-Authentication",
    "isSocial": false
  }
],
"name": "Valentyn Dominskyi",
"nickname": "vsig",
"picture": "https://avatars.githubusercontent.com/u/78111224?v=4"
```

Або ж на сайті auth0:

User Management

Users

Roles

Branding

Security

Actions






Auth Pipeline

Monitoring

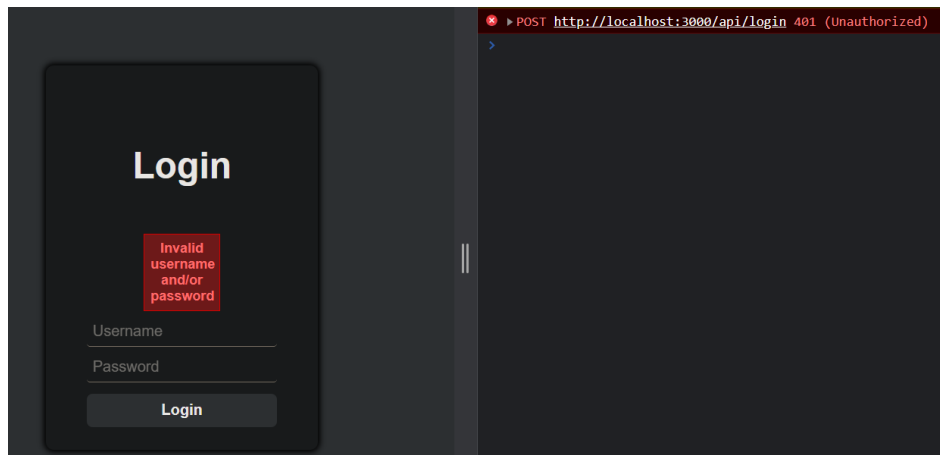
Marketplace

Extensions

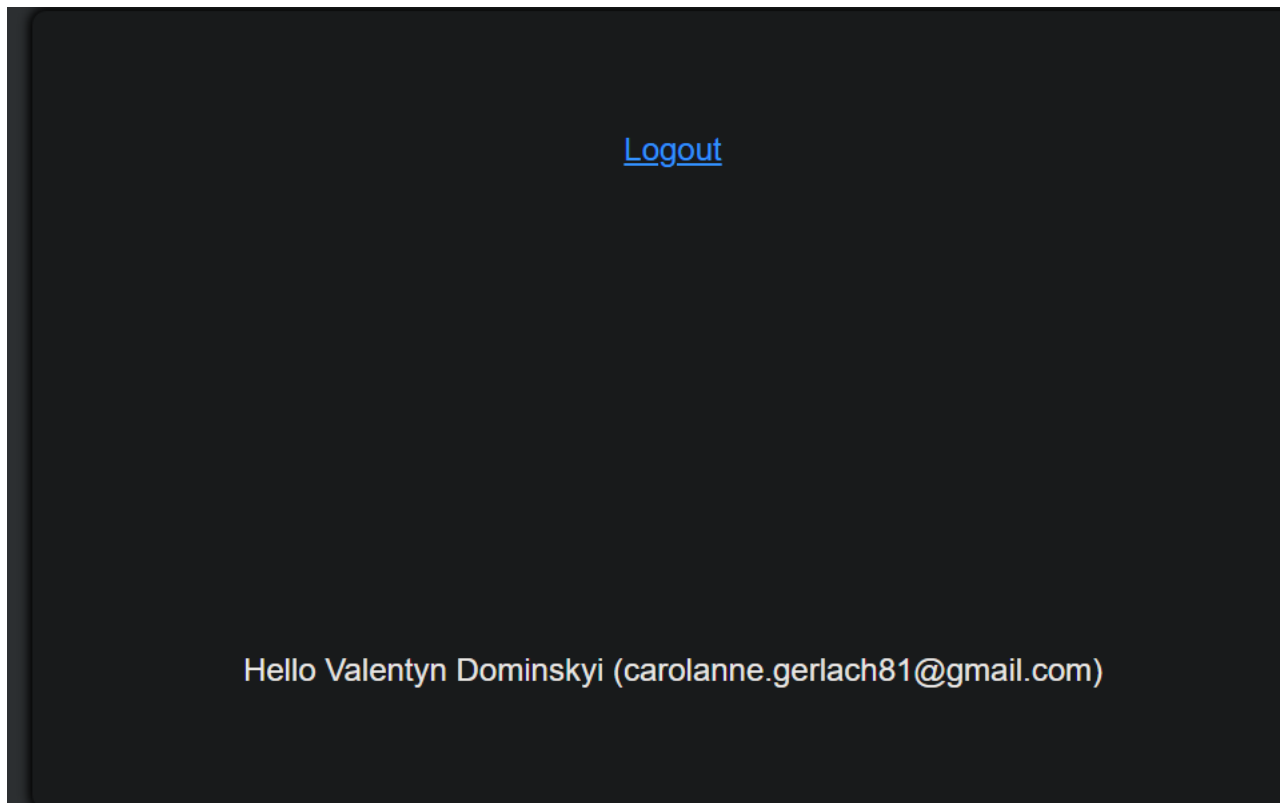
Settings

name	connection	login	last login
 <b>Valentyn Dominskyi</b> susie.hoeger64@yahoo.com	Username-Password-Authenti...	10	6 days ago
 <b>Valentyn Dominskyi</b> pablo35@gmail.com	Username-Password-Authenti...	1	9 days ago
 <b>Valentyn Dominskyi</b> belle_abbott@yahoo.com	Username-Password-Authenti...	2	9 days ago
 <b>Valentyn Dominskyi</b> ronny_leuschke@gmail.com	Username-Password-Authenti...	5	9 days ago
 <b>Valentyn Dominskyi</b> vincenzo.hoeger@gmail.com	Username-Password-Authenti...	0	never

Давайте запустимо Наш застосунок та спробуємо ввести неправильні дані:



Як видно з консолі – Нам видало помилку 401. Тепер увійдемо в Нашого користувача:



Усе пройшло успішно!

Зараз пропоную проглянути, як це все працює в коді:

- При запуску застосунку Ми намагаємося отримати токен від application:

Код:

```
app.listen(config.port, async () => {
  console.log(`Example app listening on port ${port}`);

  ⚡ const appTokenInfo = await getAppToken();
  config.appToken = appTokenInfo.access_token;
})
```

Запити:

POST ▼ https://{advanced\_domain}/oauth/token

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

☐ none ☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

	KEY	VALUE
<input checked="" type="checkbox"/>	client_id	5HdLqoxkGalwRoj3RQ9aGyGoc09oDV4V
<input checked="" type="checkbox"/>	client_secret	rTGIXeB0aJz-x2GDfmzvZbogXzZwHMuL...
<input checked="" type="checkbox"/>	audience	https://dev-wyvlhkwn475iv1wn.us.auth0...
<input checked="" type="checkbox"/>	grant_type	client_credentials

```
const appTokenOptions = {
  method: 'POST',
  url: `https://${config.domain}/oauth/token`,
  headers: {'content-type': 'application/x-www-form-urlencoded'},
  data: new URLSearchParams({
    client_id: config.clientId,
    client_secret: config.clientSecret,
    audience: config.audience,
    grant_type: 'client_credentials'
  }),
}

const getAppToken = async () => {
  try {
    const body = await axios.request(appTokenOptions);
    return body.data;
  } catch (error) {
    console.log(error);
    return null;
  }
}
```

- При спробі входу до системи потрібно взяти токен користувача за допомогою пароля та логіну, які являються підтвердженням особи

Код:

```
app.post('/api/login', async (req, res) => {
  const { login, password } = req.body;

  try {
    const userTokenInfo = await getUserToken(login, password);
    if (!userTokenInfo) {
      return res.status(401).send();
    }

    res.json({ token: userTokenInfo.access_token });
  } catch (error) {
    console.log(error)
  }
});
```

Запити:

POST

https://{{advanced\_domain}}/oauth/token

Params

Authorization

Headers (9)

Body

Pre-request Script

Tests

Setting

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

	KEY	VALUE
<input checked="" type="checkbox"/>	grant_type	http://auth0.com/oauth/grant-type/pass...
<input checked="" type="checkbox"/>	scope	offline_access
<input checked="" type="checkbox"/>	username	{{advanced_email}}
<input checked="" type="checkbox"/>	password	{{advanced_password}}
<input checked="" type="checkbox"/>	realm	Username-Password-Authentication
<input checked="" type="checkbox"/>	client_id	5HdLqoxkGalwRoj3RQ9aGyGoc09oDV4V
<input checked="" type="checkbox"/>	client_secret	rTGIXeB0aJz-x2GDfmzvZbogXzZwHMuL...
<input checked="" type="checkbox"/>	audience	https://dev-wyvlhkwn475iv1wn.us.auth0...
	Key	Value

```
const userTokenOptions = (username, password) => ({
  method: 'POST',
  url: `https://${config.domain}/oauth/token`,
  headers: {'content-type': 'application/x-www-form-urlencoded'},
  data: new URLSearchParams({
    grant_type: 'password',
    audience: config.audience,
    scope: 'offline_access',
    client_id: config.clientId,
    client_secret: config.clientSecret,
    username,
    password,
  })
});

const getUserToken = async (username, password) => {
  try {
    const options = userTokenOptions(username, password);
    const body = await axios.request(options);
    return body.data;
  } catch (error) {
    console.log(error);
    return null;
  }
}
```

- Також була змінена частина, котра пов'язана з logout:

Код:

```
app.get('/', async (req, res) => {
  if (req.userID) {
    const user = await getUser(req.userID);

    return res.json({
      username: `${user.name} (${user.email})`,
      logout: 'http://localhost:3000/logout'
    });
  }
  res.sendFile(path.join(__dirname+'index.html'));
});
```

```
const userGetOptions = (userID) => ({
  method: 'GET',
  url: `https://${config.domain}/api/v2/users/${userID}`,
  headers: {
    Authorization: `Bearer ${config.appToken}`,
  },
});

const getUser = async (userID) => {
  try {
    const options = userGetOptions(userID);
    const body = await axios.request(options);
    return body.data;
  } catch (error) {
    console.log(error);
    return null;
  }
}
```

Також було видалено масив з уже готовими користувачами, оскільки вони вже непотрібні



### **Висновок:**

Під час виконання роботи Я попрактикувався писати запити (для отримання токена користувача / застосунку та власне юзера по його ідентифікатору) не тільки в Postman (як це було у минулих роботах), а ще й за допомогою коду. Детальніше познайомився з сайтом Auth0 та його можливостями. Створив застосунок, який з ним взаємодіє

### **Посилання:**

- [Проект на GitHub](#)