Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №2

з дисципліни «Основи WEB - технологій»

Тема: «Створення системи авторизації сайту за допомогою JWT токенів»

Перевірив:
Доц. Голубєв Л. П.

Виконав:
студент групи ІМ-22
Балахон Михайло
Варіант 11

Київ - 2025

Завдання.

1. При реєстрації та при оновленні користувача пароль зберігати в зашифрованому вигляді;

2. Створити запит /users/login на вхід

3. Застосувати авторизацію: в запиті відправити authToken в заголовку Authorization. Сервер отримує токен, верифікує його і авторизує користувача (або ні в іншому випадку).

Забезпечити використання ролей (user та admin) з відповідними діями.

Хід роботи:

1. Підготовка проекту

1.1. Створення структури проекту

```
mkdir lab2

cd lab2
npm init -y
```

1.2. Встановлення необхідних пакетів

```
npm install express jsonwebtoken bcryptjs dotenv cors
```

Встановлені пакети:

- express - веб-фреймворк
- jsonwebtoken - для створення та верифікації JWT токенів
- bcryptjs - для хешування паролів
- dotenv - для роботи з змінними оточення

- cors - для налаштування CORS політики

2. Створення конфігурації

2.1. Файл .env

```
JWT_SECRET=your_super_secret_key_here_change_in_production
PORT=5000
```

3. Реалізація сервера (server.js)

3.1. Базова структура та middleware

```
require('dotenv').config();

const express = require('express');
const cors = require('cors');
const bcrypt = require('bcryptjs');
const jwt = require('jsonwebtoken');

const app = express();
app.use(cors());
app.use(express.json());
app.use(express.static('public'));
```

3.2. Сховище користувачів

Для демонстрації використовується масив в пам'яті:

```
const users = [];
```

3.3. Функція генерації JWT токена

```
const generateToken = (user) => {
  return jwt.sign(
    {
      id: user.id,
      username: user.username,
      role: user.role
    },
    process.env.JWT_SECRET,
    { expiresIn: '24h' }
  );
};
```

3.4. Middleware для перевірки токена

```
const authMiddleware = (req, res, next) => {
  const token = req.headers.authorization?.split(' ')[1];

  if (!token) {
    return res.status(401).json({ message: 'No token provided' });
  }

  try {
    const decoded = jwt.verify(token, process.env.JWT_SECRET);
    req.user = decoded;
    next();
  } catch (error) {
    return res.status(403).json({ message: 'Invalid or expired token' });
  }
};
```

3.5. Middleware для перевірки ролі адміністратора

```
const adminMiddleware = (req, res, next) => {
  if (req.user.role !== 'admin') {
    return res.status(403).json({ message: 'Access denied. Admin role required.' });
  }
  next();
};
```

4. Реалізація API endpoints

4.1. Реєстрація користувача (POST /users/register)

- Валідація вхідних даних
- Перевірка унікальності username
- Хешування пароля за допомогою bcrypt
- Створення нового користувача
- Генерація JWT токена

4.2. Вхід користувача (POST /users/login)

- Пошук користувача за username
- Перевірка пароля
- Генерація JWT токена при успішній автентифікації

4.3. Захищені маршрути

- GET /users/profile - отримання профілю поточного користувача
- GET /users - отримання списку всіх користувачів (тільки для admin)
- PUT /users/:id - оновлення користувача
- DELETE /users/:id - видалення користувача (тільки для admin)

5. Створення веб-інтерфейсу для тестування

5.1. HTML форма (public/index.html)

Створено інтерфейс з:

- Формою реєстрації (username, password, role)
- Формою входу
- Кнопками для тестування захищених endpoints
- Відображенням відповідей сервера

5.2. JavaScript логіка

- Збереження токена в localStorage
- Додавання токена до заголовків запитів
- Декодування JWT для відображення інформації про користувача
- Обробка відповідей сервера

6. Тестування системи

6.1. Реєстрація користувачів

1. Створено користувача з роллю "user"
2. Створено користувача з роллю "admin"
3. Перевірено шифрування паролів

6.2. Автентифікація

1. Вхід з правильними даними - отримання JWT токена
2. Вхід з неправильними даними - отримання помилки

6.3. Авторизація

1. Доступ до профілю - успішно з токеном
2. Доступ до списку користувачів:
    - User роль - доступ заборонено
    - Admin роль - доступ надано
3. Видалення користувача - тільки для admin

7. Безпека системи

1. Хешування паролів: Використовується bcrypt з salt rounds = 10
2. JWT токени:
    - Підписуються секретним ключем
    - Мають термін дії 24 години
    - Містять тільки необхідну інформацію (id, username, role)
3. Перевірка ролей: Додаткова перевірка для admin-only операцій
4. Валідація даних: Перевірка наявності обов'язкових полів

Структура проекту

```
lab2/
├── server.js          # Основний сервер з API
├── .env               # Змінні оточення
├── package.json       # Конфігурація проекту
├── public/
│   └── index.html     # Веб-інтерфейс для тестування
└── README.md          # Документація
```

Результати роботи

Створено повнофункціональну систему автентифікації та авторизації з:

1. Безпечним зберіганням паролів - всі паролі хешуються перед збереженням
2. JWT автентифікацією - токени генеруються при вході та реєстрації
3. Рольовою моделлю - розмежування прав доступу між user та admin
4. Захищеними endpoints - перевірка токена та ролей
5. Веб-інтерфейсом - для зручного тестування API

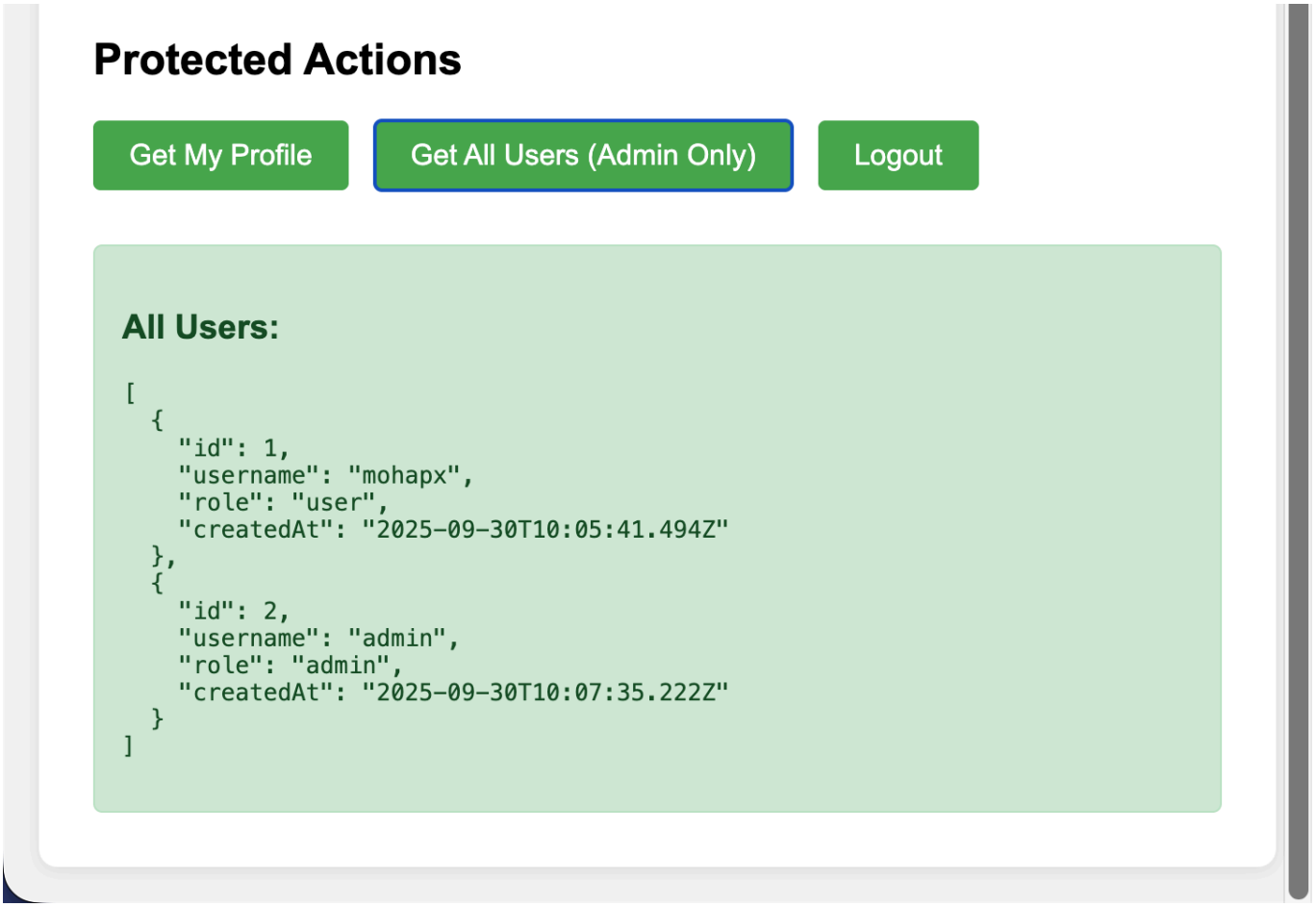
Висновки

В ході виконання лабораторної роботи:

- Вивчено принципи роботи JWT токенів
- Реалізовано безпечне зберігання паролів з використанням bcrypt
- Створено систему автентифікації та авторизації
- Впроваджено рольову модель доступу
- Отримано практичні навички створення захищених API

JWT токени є ефективним рішенням для створення stateless автентифікації в сучасних веб-додатках, забезпечуючи безпеку та масштабованість системи.

Знімки екрана:



JWT Authentication Demo

Register

Username:

admin

Password:

••••••

Role:

Admin

Register

Success! User registered successfully

```
Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MiwidXNlcm5hbWUiOiJhZ
G1pbiIsInJvbGUiOiJhZG1pbiIsImlhdCI6MTc1OTIyNjg1NSwiZXhwIjoxNzU5MzEzMjU1f
Q.lPyaVaJzXaX_PtU5Q3sJ4HW773TyocEAQ_zPj_UAEJg
```

Login

Username:

Password:

Login

Logged in as:

Username: admin

Role: admin

User ID: 2