Dokumentacja testów API Event API

(Lista 4, Nowoczesne technologie WWW)

1. Parametry wstępne

- **Technologie**: Express.js, MongoDB
- **Narzędzie do testów**: Postman
- **Lokalizacja**: http://localhost:3000/api

2. Opis testów

2.1 Tworzenie administratora

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/users
- **Body (JSON):**
  username: "testadmin"
  password: "test123"
  role: "admin"
- **Wynik:** Status 201 Created, użytkownik testadmin utworzony jako administrator.

2.2 Logowanie administratora

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/auth/login
- **Body (JSON):**
  username: "testadmin"
  password: "test123"
- **Wynik:** Status 200 OK, zwrócony token JWT.

2.3 Pobieranie wszystkich użytkowników (jako admin)

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/users
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Wynik:** Status 200 OK, lista użytkowników.

2.4 Usuwanie nieistniejącego użytkownika (jako admin)

- **Metoda:** DELETE
- **Endpoint:** http://localhost:3000/api/users/681f379eda9aba7b6ce262ab
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Wynik:** Status 404 Not Found, komunikat o braku użytkownika.

2.5 Tworzenie użytkownika do testu usuwania

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/users
- **Body (JSON):**
  username: "testuserfordeletion"
  password: "test123"
  role: "user"
- **Wynik:** Status 201 Created, użytkownik utworzony.

2.6 Usuwanie istniejącego użytkownika (jako admin)

- **Metoda:** DELETE
- **Endpoint:** http://localhost:3000/api/users/681f9ac1b9c7facc82cd58c7
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Wynik:** Status 204 OK, użytkownik został usunięty.

2.7 Tworzenie wydarzenia (jako admin)

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/events
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Body (JSON):**
  title: "Dev Meetup"
  description: "Local developers' networking event"
  date: "2025-09-10T18:00:00Z"
- **Wynik:** Status 201 Created, wydarzenie zostało utworzone.

2.8 Pobieranie wydarzeń (jako admin)

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/events?page=1&limit=5&sortBy=date
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Wynik:** Status 200 OK, lista wydarzeń posortowana po dacie.

2.9 Tworzenie użytkownika

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/users
- **Body (JSON):**
  username: "testuser"
  password: "test123"
  role: "user"
- **Wynik:** Status 201 Created, użytkownik został utworzony.

2.10 Logowanie użytkownika

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/auth/login
- **Body (JSON):**
  username: "testuser"
  password: "test123"
- **Wynik:** Status 200 OK, zwrócony token JWT.

2.11 Pobieranie wydarzeń (jako użytkownik)

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/events?page=1&limit=5&sortBy=date
- **Nagłówek:** Authorization: Bearer <valid-user-token>
- **Wynik:** Status 200 OK, lista wydarzeń.

2.12 Pobieranie użytkowników (jako użytkownik)

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/users
- **Nagłówek:** Authorization: Bearer <valid-user-token>
- **Wynik:** Status 403 Forbidden, brak uprawnień.

2.13 Usuwanie użytkownika (jako użytkownik)

- **Metoda:** DELETE
- **Endpoint:** http://localhost:3000/api/users/681f38e7da9aba7b6ce262b1
- **Nagłówek:** Authorization: Bearer <valid-user-token>
- **Wynik:** Status 403 Forbidden, brak uprawnień.

2.14 Próba utworzenia użytkownika o istniejącej nazwie

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/users
- **Nagłówek:** Authorization: Bearer <valid-user-token>
- **Body (JSON):**
  username: "testuser"
  password: "test456"
  role: "user"
- **Wynik:** Status 409 Conflict, użytkownik o takiej nazwie już istnieje.

2.15 Próba logowania z błędnym hasłem

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/auth/login
- **Body (JSON):**
  username: "testuser"
  password: "wrongpassword"

- **Wynik:** Status 401 Unauthorized, komunikat o nieprawidłowych danych logowania.

2.16 Próba pobrania użytkowników bez autoryzacji

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/users
- **Wynik:** Status 401 Unauthorized, brak tokena autoryzacyjnego.

2.17 Próba pobrania użytkowników z nieprawidłowym tokenem

- **Metoda:** GET
- **Endpoint:** http://localhost:3000/api/users
- **Nagłówek:** Authorization: Bearer <invalid-token>
- **Wynik:** Status 401 Unauthorized, nieprawidłowy token.

2.18 Próba utworzenia użytkownika bez podania hasła

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/users
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Body (JSON):**
  username: "nopassworduser"
  role: "user"
- **Wynik:** Status 400 Bad Request, nazwa użytkownika i hasło są wymagane.

2.19 Próba utworzenia wydarzenia z niewłaściwym formatem daty

- **Metoda:** POST
- **Endpoint:** http://localhost:3000/api/events
- **Nagłówek:** Authorization: Bearer <valid-admin-token>
- **Body (JSON):**
  title: "Invalid Event"
  description: " This event has an invalid date"
  date: " invalid-date"
- **Wynik:** Status 400 Bad Request, nieprawidłowa data.

2.20 Próby wysyłania requestów z tokenami wykorzystującymi różne algorytmy

- W API dopuszczam tylko **HS256**
- Tokeny tworzyłam na stronie jwt.io i ustawiałam jako cookies w narzędziu Postman
- Próby – wyświetlanie użytkowników, tworzenie eventów:
    - Algorithm **‘none’** + admin jako role w payload
    - Algorithm **‘none’** + user jako role w payload
    - Algorithm **‘RS256’** + admin jako role w payload
    - Algorithm **‘RS256’** + user jako role w payload
    - Algorithm **‘RS256’** + admin jako role w payload (usunięty podpis)
    - Algorithm **‘RS256’** + user jako role w payload (usunięty podpis)
- **Wynik wszystkich prób:** Status 401 Unauthorized