

Spis Treści

API HackYeah	2
API Szybki Tutorial	3
Endpointy	4
Tworzenie użytkownika	5
Usuwanie użytkownika	7
Znajdywanie danych (publicznych) użytkownika	9
Znajdywanie danych (publicznych) wszystkich użytkowników	11
Logowanie	13
Wylogowywanie	15
Edytowanie nazwy użytkownika	17
Pobieranie informacji (prywatnych) użytkownika	19

API HackYeah

Wprowadzenie

W tym pliku PDF znajdziesz wszystkie potrzebne endpointy i jak używać tego API

Base URL - Podstawowe URL

IP4_Adress:8765/api/

API Szybki Tutorial

Znajdziesz tutaj poradnik krok po kroku jak obsługiwać to API

Authentication - Uwierzytelnianie

Sposobem uwierzytelniania jest podawanie tokenu w ciele zapytania. Token będzie zwracany, podczas logowania

Wysyłanie żądań

```
GET /api/users/test HTTP/1.1
Host: IP4_Adress:8765
```


Musi zwrócić status: 200

Response Handling - Obsługa odpowiedzi

```
status: 200 - OK
status: 201 - Zapisano wysłany zasób
status: 400 - Nie prawidłowe zapytanie
status: 401 - Nie autoryzowany dostęp
status: 403 - Zabroniony dostęp do zasobów
status: 404 - Nie znaleziono zasobów
status: 405 - Nie dozwolona metoda
status: 500 - Wewnętrzny błąd serwera
```

Endpointy

Tutaj znajdziesz wszystkie endpointy

 Tutaj znajdziesz wszystko na temat API.

Tworzenie użytkownika

POST /users

Tworzenie użytkownika

Request parameters

Query

username object **required**

Nazwa

Child attributes

email object **required**

Email

Child attributes

password object **required**

Hasło

Child attributes

two_factor object

Podwójne uwierzytelnianie (0 lub 1)

Child attributes

JSON example

```
{
  "username": "Waleń",
  "email": "LubimyHejj@pie.pl",
  "password": "Super_tajne_hasło"
```

```
"two_factor": 1 (or 0)
}
```

Responses

201 OK

Content type

OK

201

```
{
  "message": "User has been created"
}
```

400 Złe zapytanie

Content type

Złe zapytanie

400

```
{
  "message": "User has not been created"
}
```

Usuwanie użytkownika

DELETE /users/{id}

Możesz usunąć użytkownika jedynie gdy posiada token

Request parameters

Path

id object **required**

Identyfikator użytkownika

Child attributes

Header

token object **required**

Token uwierzytelniania użytkownika

Child attributes

HTTP

GET /api/users/{username} HTTP/1.1

Host: IP4_Adress:8765

Authorization: {token}

Responses

200 Znaleziono użytkownika

Content type

Znaleziono użytkownika

200

```
{  
  "message": "User has been deleted"  
}
```

400 Zły token/zapytanie

Content type

Zły token/zapytanie

400

```
{  
  "message": "Invalid token or query"  
}
```

404 Nie znaleziono użytkownika

Content type

Nie znaleziono użytkownika

404

```
{  
  "message": "Invalid id"  
}
```


Znajdywanie danych (publicznych) użytkownika

GET /users/{id}

Request parameters

Path

id object **required**

Identyfikator użytkownika

Child attributes

Responses

200 OK

Content type

OK

200

```
{
  "id": 2137,
  "username": "Waleń",
  "points": 420
}
```

400 Źle skonstruowane zapytanie

Content type

Źle skonstruowane zapytanie

400

```
{  
  "message": "Invalid Query"  
}
```

404 Nie znaleziono użytkownika

Content type

Nie znaleziono użytkownika

404

```
{  
  "message": "User has not been found"  
}
```

Znajdywanie danych (publicznych) wszystkich użytkowników

GET /users

Pobieranie wszystkich danych publicznych użytkowników

Responses

200 Znaleziono użytkowników

Content type

Znaleziono użytkowników

200

```
[
  {
    "id": 2137,
    "username": "Waleń",
    "points": 420
  }
]
```

400 Zły token/zapytanie

Content type

Zły token/zapytanie

400

```
{
  "message": "Invalid Query"
}
```

404 Brak użytkowników

Content type

Brak użytkowników

404

```
{  
  "message": "Users have not been found"  
}
```

Logowanie

POST /login

Logowanie do systemu

Request parameters

Query

username object **required**

Nazwa użytkownika

Child attributes

password object **required**

Hasło

Child attributes

Responses

200 OK

Content type

OK

X-Expires-After (Header) object **required**

Data kiedy token wygasa (UTC)

Child attributes

200

```
{  
  "id": 1,
```

```
"username" : "Waleń",  
"email" : "lubimyHejj@pie.pl",  
"points": 420,  
"token": "123217eg27q"  
}
```

400 Źle skonstruowane zapytanie

Content type

Źle skonstruowane zapytanie

400

```
{  
  "message": "Invalid query"  
}
```

404 Zła nazwa użytkownika/Złe hasło

Content type

Zła nazwa użytkownika/Złe hasło

404

```
{  
  "message": "Invalid username or password"  
}
```

Wylogowywanie

DELETE /login

Wylogowywanie

Request parameters

Header

token object **required**

Token uwierzytelniania

Child attributes

HTTP

GET /api/users/{username} HTTP/1.1

Host: IP4_Adress:8765

Authorization: {token}

Responses

200 OK

Content type

OK

200

```
{
  "message": "User has been logged out"
}
```

400 Źle skonstruowane zapytanie

Content type

Źle skonstruowane zapytanie

400

```
{  
  "message": "Invalid query"  
}
```

404 Źle podany token

Content type

Źle podany token

404

```
{  
  "message": "Invalid token"  
}
```


Edytowanie nazwy użytkownika

PUT /login/{username}

Możesz tylko zmienić użytkownika, gdy podasz token

Request parameters

Header

token object **required**

token uwierzytelniania

Child attributes

Path

username object **required**

Nazwa użytkownika

Child attributes

HTTP

GET /api/login/{username} HTTP/1.1

Host: IP4_Adress:8765

Authorization: {token}

Responses

201 Zedytowano użytkownika

Content type

Zedytowano użytkownika

201

```
{  
  "message": "User has been edited"  
}
```

400 Źle skonstruowane zapytanie

Content type

Źle skonstruowane zapytanie

400

```
{  
  "message": "Invalid query"  
}
```

404 Zła nazwa użytkownika

Content type

Zła nazwa użytkownika

404

```
{  
  "message": "Invalid username"  
}
```

Pobieranie informacji (prywatnych) użytkownika

GET /login/{username}

Request parameters

Header

token object **required**

Token uwierzytelniania

Child attributes

HTTP

```
GET /api/users/{username} HTTP/1.1
Host: IP4_Adress:8765
Authorization: fwhfh27g6dgqw
```

Responses

200 OK

Content type

OK

200

```
{
  "id": 1,
  "username": "Waleń",
  "email": "LubimyHejj@pie.pl",
```

```
"verified": 1 (or 0)
}
```

400 Źle skonstruowane zapytanie/Zły token

Content type

Źle skonstruowane zapytanie/Zły token

400

```
{
  "message": "Invalid query or token"
}
```

404 Nie znaleziono użytkownika

Content type

Nie znaleziono użytkownika

404

```
{
  "message": "Invalid username"
}
```