Krzysztof Kogut

Wspólna Kasa

API

Spis treści

[Wstęp 3](#_Toc419808164)

[Account 4](#_Toc419808165)

[POST: Token 4](#_Toc419808166)

[GET: /api/Account/UserInfo 4](#_Toc419808167)

[POST: /api/Account/Logout 4](#_Toc419808168)

[GET: /api/Account/ManageInfo?returnUrl=%2F&generateState=true 4](#_Toc419808169)

[POST: /api/Account/ChangePassword 5](#_Toc419808170)

[POST: /api/Account/SetPassword 5](#_Toc419808171)

[POST: /api/Account/AddExternalLogin 5](#_Toc419808172)

[POST: /api/Account/RemoveLogin 5](#_Toc419808173)

[GET: /api/Account/ExternalLogins?returnUrl=%2F&generateState=true 5](#_Toc419808174)

[POST: /api/Account/Register 5](#_Toc419808175)

[POST: /api/Account/RegisterExternal 5](#_Toc419808176)

[Group 7](#_Toc419808177)

[GET: /api/Groups 7](#_Toc419808178)

[GET: /api/Groups/5 7](#_Toc419808179)

[PUT: /api/Groups/5 7](#_Toc419808180)

[POST: /api/Groups 7](#_Toc419808181)

[DELETE: /api/Groups/5 7](#_Toc419808182)

[Expense 8](#_Toc419808183)

[GET: /api/Expenses 8](#_Toc419808184)

[GET: /api/Expenses/5 8](#_Toc419808185)

[PUT: /api/Expenses/5 8](#_Toc419808186)

[POST: /api/Expenses 8](#_Toc419808187)

[DELETE: /api/Expenses/5 8](#_Toc419808188)

[Transfer 9](#_Toc419808189)

[GET: /api/Transfers 9](#_Toc419808190)

[GET: /api/Transfers/5 9](#_Toc419808191)

[PUT: /api/Transfers/5 9](#_Toc419808192)

[POST: /api/Transfers 9](#_Toc419808193)

[DELETE: /api/Transfers/5 9](#_Toc419808194)

[Typu obiektów i ich struktura 10](#_Toc419808195)

[Obiekt typu Group 10](#_Toc419808196)

[Obiekt typu Expense 10](#_Toc419808197)

[Obiekt typu Transfer 10](#_Toc419808198)

[Obiekt typu ApplicationUser 10](#_Toc419808199)

# Wstęp

W tym dokumencie zostaną opisane metody API[[1]](#footnote-1), za pomocą których aplikacja mobilna będzie mogła wykonywać operacje w systemie Wspólna Kasa.

Wszystkie żądania należy kierować protokołem HTTP[[2]](#footnote-2) pod adres <http://wspolnakasa.azurewebsites.net/>, dodając podane niżej sufiksy. Żądania mogą być typu: GET, POST, PUT albo DELETE, co zostanie określone.

Do opisywania przesyłanych obiektów, zostanie użyty standard JSON[[3]](#footnote-3).

# Account

## POST: /Token

Zwraca access\_token, którego używa się do wysyłania żądań wymagających autoryzacji. Należy wysłać obiekt:

{

“grant\_type”: ”password”,

“username”: “login121@gmail.com”,

“password”: “pass123”

}

W odpowiedzi, zostanie zwrócony obiekt:

{

"access\_token": "5SEwL\_pEI4flvV7oWgO41zNH8XSZ9pVmJ7CdPgmg5jryH6wJjHU4FPsIsZNyc8r\_...",

"token\_type": "bearer",

"expires\_in": 1209599,

"userName": " login121@gmail.com",

".issued": "Tue, 19 May 2015 11:59:20 GMT",

".expires": "Tue, 02 Jun 2015 11:59:20 GMT"

}

Każde żądanie, które wymaga autoryzacji, musi być wysłane z parametrem **Authorization** o wartości bearer 5SEwL\_pEI4flvV7oWgO41zNH8XSZ9pVmJ7CdPgmg5jryH6wJjHU4FPsIsZNyc8r\_...., gdzie po słowie *bearer* wpisujemy otrzymany wcześniej access\_token.

## GET: /api/Account/UserInfo

Zwraca informację o bieżącym użytkowniku w postaci:

{

“email”: ”name@domain.com”,

“hasRegistered”: false,

“loginProvider”: “Facebook”

}

## POST: /api/Account/Logout

Wylogowuje bieżącego użytkownika z systemu.

## GET: /api/Account/ManageInfo?returnUrl=%2F&generateState=true

Zwraca informację o dostępnych sposobach logowania dla bieżącym użytkownika w postaci:

{

"LocalLoginProvider": "Local",

"Email": "email@domain.com",

"Logins": [

{

"LoginProvider": "Facebook",

"ProviderKey": "4830681171006466"

}

],

"ExternalLoginProviders": [

{

"Name": "Facebook",

"Url": "/api/Account/ExternalLogin?provider=Facebook&response\_type=token&client\_id=self&redirect\_uri=...",

"State": "6uGtoHg8-tH0O1shVUbvxdOhY0TweRFGBofqG5fXTWA1"

}

]

}

## POST: /api/Account/ChangePassword

Zmienia hasło bieżącego użytkownika. Należy przesłać obiekt:

{

“oldPassword”: “pass-123”,

“newPassword”: “123-pass”,

“confirmPassword”: “123-pass”,

}

## POST: /api/Account/SetPassword

Ustawia po raz pierwszy hasło dla bieżącego użytkownika. Należy przesłać obiekt:

{

“newPassword”: “123-pass”,

“confirmPassword”: “123-pass”,

}

## POST: /api/Account/AddExternalLogin

Dodaje dostawcę logowania zewnętrznego do istniejącego użytkownika. Należy przesłać obiekt:

{

“externalAccessToken”: “QotufgXRptkAfJvcthIOWBnGZydgVkZWsx8YrQepeDk1”

}

## POST: /api/Account/RemoveLogin

Usuwa dostawcę logowania zewnętrznego od istniejącego użytkownika. Należy przesłać obiekt:

{

“loginProvider”: “Facebook”,

“providerKey”: “4830681171006466”

}

## GET: /api/Account/ExternalLogins?returnUrl=%2F&generateState=true

Zwraca listę wszystkich dostępnych sposobów logowania w postaci listy obiektów postaci:

{

“name”: “Provider”,

“url”: “/api/Account/ExternalLogin?provider=...”,

“state”: “QotufgXRptkAfJvcthIOWBnGZydgVkZWsx8YrQepeDk1”

}

## POST: /api/Account/Register

Rejestruje użytkownika w systemie Wspólna Kasa. Należy przesłać obiekt:

{

“email”: “name@domain.com”,

“password”: “123-pass”,

“confirmPassword”: “123-pass”,

“userName”: “name”

}

Pole userName będzie używane do wyświetlania nazwy użytkownika w aplikacji Wspólna Kasa.

## POST: /api/Account/RegisterExternal

Rejestruje użytkownika zalogowanego przez zewnętrznego dostawcę logowania w systemie Wspólna Kasa. Należy przesłać w tym żądaniu obiekt:

{

“email”: “name@domain.com”,

“userName”: “name”

}

Pole userName będzie używane do wyświetlania nazwy użytkownika w aplikacji Wspólna Kasa.

# Group

## GET: /api/Groups

Zwraca listę grup.

## GET: /api/Groups/5

Zwraca konkretną grupę. (Za 5 podstawiamy odpowiednie ID).

## PUT: /api/Groups/5

Aktualizuje grupę. W ciele żądania należy przesłać obiekt typu Group. W odpowiedzi zwraca zaktualizowaną grupę.

## POST: /api/Groups

Dodaje grupę. W ciele żądania należy przesłać obiekt typu Group. W odpowiedzi zwraca dodaną grupę.

## DELETE: /api/Groups/5

Usuwa grupę o podanym ID. W odpowiedzi zwraca usuniętą grupę.

# Expense

## GET: /api/Expenses

Zwraca listę wydatków.

## GET: /api/Expenses/5

Zwraca konkretny wydatek. (Za 5 podstawiamy odpowiednie ID).

## PUT: /api/Expenses/5

Aktualizuje wydatek. W ciele żądania należy przesłać obiekt typu Expense. W odpowiedzi zwraca zaktualizowany wydatek.

## POST: /api/Expenses

Dodaje wydatek. W ciele żądania należy przesłać obiekt typu Expense. W odpowiedzi zwraca dodany wydatek.

## DELETE: /api/Expenses/5

Usuwa wydatek o podanym ID. W odpowiedzi zwraca usunięty wydatek.

# Transfer

Pole applicationUserId to ID użytkownika, który wykonał przelew, a receiverId to ID odbiorcy przelewu.

Pole date jest typu DateTime.

## GET: /api/Transfers

Zwraca listę przelewów.

## GET: /api/Transfers/5

Zwraca konkretny przelew. (Za 5 podstawiamy odpowiednie ID).

## PUT: /api/Transfers/5

Aktualizuje przelew. W ciele żądania należy przesłać obiekt typu Transfer. W odpowiedzi zwraca zaktualizowany przelew.

## POST: /api/Transfers

Dodaje przelew. W ciele żądania należy przesłać obiekt typu Transfer. W odpowiedzi zwraca dodany przelew.

## DELETE: /api/Transfers/5

Usuwa przelew o podanym ID. W odpowiedzi zwraca usunięty przelew.

# Typu obiektów i ich struktura

## Obiekt typu Group

{

“groupId”: 5,

“name”: “nazwa”,

“secret”: “haslo”,

“members”: [ lista członków grupy ],

“transfers”: [ lista przelew wewnątrz grupy ],

“expenses”: [ lista wydatków wewnątrz grupy ]

}

## Obiekt typu Expense

{

“expenseId”: 5,

“userPayingId”: “id-uzytkownika”,

“description”: “opis”,

“date”: 1431547297994,

“participants”: [ lista użytkowników partycypujących w wydatku ],

“amount”: 5.19,

“groupId”: 1,

}

## Obiekt typu Transfer

{

“transferId”: 5,

“applicationUserId”: “id-uzytkownika”,

“groupId”: 1,

“receiverId”: “id-odbiorcy”,

“description”: “tytul przelewu”,

“amount”: 5.19,

“date”: 1431547297994

}

## Obiekt typu ApplicationUser

{

“displayName”: “nazwa uzytkownika”,

“email”: “mail@domain.com”,

“emailConfirmed”: true,

“id”: “id-uzytkownika”

}

1. Interfejs programistyczny aplikacji [↑](#footnote-ref-1)
2. HyperText Transfer Protocol [↑](#footnote-ref-2)
3. JavaScript Object Notation [↑](#footnote-ref-3)