**CHESSIN**

REST API DOCUMENTATION

1. **Overview**

REST API allows users to create and manage their accounts and is also used to service chess games. It allows the following actions in :

* Account management (public):
  + Register new user,
  + Authenticate (login),
  + Change password,
  + Remind password,
  + Refresh access token,
  + Verify verification code,
  + Check whether user has two factor authentication enable,
* Account management (private):
  + Get user details by email
* Chess games management (private):
  + Search for new game

1. **Register**

First, the register request takes user details and creates new, inactive user in the database. After that, verification code is sent to the given email address.

*POST api/v1/auth/register*

{

    "firstname": "",

    "lastname": "",

    "email": "",

    "nameInGame": "",

    "password": "",

    "country": ""

}

Then, to activate the account, user needs to verify the verification code. After successful activation, user receives access and refresh token.

*POST api/v1/auth/verifyCode*

{

    "email": "",

    "verificationCode": "",

    "verificationType": "REGISTER"

}

Note the verificationType attribute: it is used to specify code verification purpose. Possible verification types:

*REGISTER*,  
*AUTHENTICATE*,  
*CHANGE\_PASSWORD*,  
*REMIND\_PASSWORD*,  
*CHANGE\_EMAIL*

1. **Authenticate (login)**

First, client should check whether user has two factor authentication enabled.

*POST api/v1/auth/2faEnabled*

{

    "email": "",

}

* 1. **Without 2fa enabled**

Client sends authentication request and, if successful, receives access and refresh token.

*POST api/v1/auth/authenticate*

{

    "email": "",

    "password": ""

}

* 1. **With 2fa enabled**

Client sends authentication request with only email attribute set.

*POST api/v1/auth/authenticate*

{

    "email": "",

}

Then, server sends verification code to the given email address. After that, client sends the verification code with email and password attributes set.

*POST api/v1/auth/verifyCode*

{

    "email": "",

    "password": "",

    "verificationCode": "",

    "verificationType": "AUTHENTICATE"

}

1. **Change password**

Identically to the authentication process, client should check whether user has 2fa enabled.

*POST api/v1/auth/2faEnabled*

{

    "email": "",

}

* 1. **Without 2fa enabled**

Client sends change password request with all attributes set.

*POST api/v1/auth/changePassword*

{

    "email": "",

    "oldPassword": "",

    "newPassword": ""

}

* 1. **With 2fa enabled**

Client sends change password request with only email attribute set.

*POST api/v1/auth/changePassword*

{

    "email": ""

}

After receiving verification code, client sends verification code request with old password, new password and verification code.

*POST api/v1/auth/verifyCode*

{

    "email": "",

    "password": "",

    "newPassword": "",

    "verificationCode": "",

    "verificationType": "CHANGE\_PASSWORD"

}

1. **Remind password**

Reminding password always requires two factor authentication regardless of account settings. Therefore, client sends remind password request with email and new password attributes set – sending the new password in the first request is required to ensure that the password is different from the old password and also to allow client application to require new password confirmation in the second request in order to ensure that the user remembers the new password.

*POST* api/v1/auth/remindPassword

{

    "email": "",

    "newPassword": ""

}

Then, after successful request, client sends received verification code together with the new password attribute set.

{

    "email": "",

    "newPassword": "",

    "verificationCode": "",

    "verificationType": "REMIND\_PASSWORD"

}

1. **Refresh token**

By default, access token has 15 minute expiration time. After that, user has to renew the token by sending token refresh request.

*POST api/v1/auth/refreshToken*

{

    "refreshToken": ""

}

After successful verification, server return the newly generated access token with another 15 minute of expiration time.