Derek Sun

CS 496 Final Project

Link: https://finalproject-206401.appspot.com/

Link directs to a page to initiate an Oauth2.0 request to access user's information through the Google Plus API. It will create the user in the database using a POST request and assign the user a new access token. If the user already exists, identified through their Google Plus URL, then only a new access token is assigned and nothing else is altered through a PUT request.

API: "Data provided for free by IEX. View IEX's Terms of Use."
API Link: "https://api.iextrading.com/1.0"
API Description: The API provides stock data which is used in this project in order to gather data for stock prices.

Entities:
User:
- id: String Property
- userURL: String Property
- access_token: String Property
- fname: String Property
- lname: String Property
- email: String Property
- portfolio: String Property

Portfolio:
- id: String Property
- name: String Property
- owner: String Property
- stock_portfolio: Structured Property
- new_worth: Float Property

Stock entity is required since stock_portfolio is a structured property
Stock:
- ticker: String Property
- amount: Integer Property

POST /user
Description: Creates a User entity with the following specifications and returns the user info.
- userURL: String Property
- access_token: String Property
- fname: String Property
- lname: String Property
- email: String Property

Status Code: 201 if successful, 400 if post data is invalid

GET /user
Description: Returns an array of all existing users in the database. The entities given are id, fname, and lname.
Status Code: 200 if successful

GET /user/{userid}

Description: Gets the information of the requested user. Requires access token.
Status Code: 200 if successful, 404 if user does not exist, 403 if access token missing or incorrect.

PUT /user/{userid}
Description: Used for Oauth2.0 access to update the user access token if the User already exists. Returns the User info.
Status Code: 200 if successful, 404 if user does not exist, 400 if data is invalid

DELETE /user/{userid}
Description: Deletes User from database and removes it as owner from all associated Portfolios. Requires access token.
Status Code: 204 if successful, 404 if user does not exist, 403 if access token is missing or incorrect.

PATCH /user/{userid}
Description: Changes the user data. Requires access token. The following are allowed to be changed.
    access_token: String Property
    fname: String Property
    lname: String Property
    email: String Property
Status Code: 200 if successful, 404 if user does not exist, 403 if access token missing or incorrect.

POST /portfolio
Description: Creates a new portfolio in the database with the following specification.
    name: String Property
    stock_portfolio: String Property (Can be empty list or list with JSON entities “ticker” and “amount”)
    ex: [{“ticker”: “A”, “amount”: 10}] denotes stock_portfolio has 10 stocks of company “A”
Status Code: 201 if successful, 400 if data is incorrect

GET /portfolio
Description: Returns a list of all the portfolios in the database. Data from portfolio given is the owner and portfolio ID.
Status Code: 200 if successful

GET /portfolio/{portfolioid}
Description: Gets the portfolio data from the database. If the portfolio belongs to a user, then access token is required.
Status Code: 200 if successful, 404 if portfolio does not exist, 403 if access token is missing or incorrect.

DELETE /portfolio/{portfolioid}
Description: Deletes the portfolio data from the database. If the portfolio belongs to a user, then access token is required and the portfolio is removed from the user’s portfolio.
Status Code: 204 if successful, 404 if portfolio does not exist, 403 if access token is missing or incorrect.

PATCH /portfolio/{portfolioid}
Description: Updates the portfolio data from the database. If the portfolio belongs to a user, then access token is required.
Status Code: 204 if successful, 404 if portfolio does not exist, 403 if access token is missing or incorrect, 400 if data is invalid.

PUT /user/{userid}/portfolio/{portfolioid}
Description: Assigns a portfolio to the user. The portfolio owner is updated.
Status: 204 if successful, 404 if portfolio or user does not exist, 403 if access token is missing, incorrect, or if portfolio is already taken

DELETE /user/{userid}/portfolio/{portfolioid}
Description: Removes the user as the owner of the portfolio. The portfolio is also removed from the list of portfolios from the user.
Status: 204 if successful, 404 if portfolio or user does not exist, 403 if access token is missing, incorrect, or if portfolio does not belong to the user

PATCH /user/{userid}/portfolio/{portfolioid}
Description: Updates the Portfolio data once it has been taken by a user.
Status: 200 if successful, 404 if portfolio or user does not exist, 403 if access token is missing or incorrect, 400 if data is invalid

GET /user/{userid}/portfolio/{portfolioid}
Description: Gets the Portfolio information from the data base.
Status: 200 if successful, 404 if portfolio or user does not exist, 403 if access token is missing or incorrect, 400 if data is invalid

GET /clearDB
Description: Deletes Everything From the Database
Status: 200 if Successful