API Specification (v0.0.1)

Written by Aaron Vontell for VP Technologies

This document contains the specification for the VP Technologies fitness application. The root URL for all of these requests is the following:

https://vptech-fitness.herokuapp.com

The API is split into two parts; an unsecured API (which starts with path /api), and a secured API (which starts with path /apis). The protected endpoint essentially require an Access Token for a specific user.
For ease of access, use the Doc outline (enabled in Tools > Document outline).

When implemented on Android, please highlight the title in green.

User API

The following are endpoints for creating, updating, and accessing users.

Create User

ENDPOINT: /api/users
HTTP REQUEST: POST
HEADERS: None

BODY

- username VARCHAR (string)
- email VARCHAR (string)
- name VARCHAR (string)
- password VARCHAR (string)

RESPONSE

- status VARCHAR (string) - either “success” or “failure”
- message VARCHAR (string) - a message for the success/failure

NOTES

- A 500 error will be thrown, and status will be “failure” in two cases
  - The user already exists
  - One of the fields above is missing (i.e. username, email, name, and password are all required)

EXAMPLE

Request: POST https://vptech-fitness.herokuapp.com/api/users

Body:

```
{
      username: “vontell”,
      email: “vontell@mit.edu”,
      name: “Aaron Vontell”,
      password: “passwordtest”
}
```

Response:

```
{
      status: “success”,
      message: “Inserted new user vontell”
}
```

Authenticate (i.e. Login In)

ENDPOINT: /oauth/token

HTTP REQUEST: POST

HEADERS:

- Authorization: “Basic dnBmaXR3ZWJhcHA6SkJVWTlWRTY5MjQzQllDOTAyNDM4N0hHVlkzQVFGSw==”

BODY

- username VARCHAR (string)
- password VARCHAR (string)
- grant_type “password” -- This should always be “password”

RESPONSE (200 OK)

- token_type VARCHAR (string)
- access_token VARCHAR (string) - Their login key. THIS IS IMPORTANT
- expires_in NUMBER

RESPONSE (500 ERROR)

- code NUMBER
- error VARCHAR (string) - The type of error
- error_description VARCHAR (string) - server_error

NOTES

- This endpoint essentially returns an access token which is used to authenticate all other requests. The “access_token” response needs to be saved on the device! This will be used later.
- If a 500 error occurs (i.e. access_token field is not present) then the user should be shown an error which is something along the lines of “Username or password are incorrect”.
- Don’t forget about the header!

EXAMPLE

Request: POST https://vptech-fitness.herokuapp.com/oauth/token
Body:

```
{
      username: “vontell”,
      password: “passwordtest”,
      grant_type: “password”
}
```

Response:

```
{
      token_type: "bearer",
      access_token: “d62133269159968f54241b39feecef31869bdc94”,
      Expires_in: 3600
}
```

Validate (i.e. Is user logged in?)

ENDPOINT: /oauth/validate
HTTP REQUEST: GET
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- None

RESPONSE (200 OK)

- authorized true (BOOLEAN)

RESPONSE (500 ERROR)

- code NUMBER
- error VARCHAR (string) - The type of error
- error_description VARCHAR (string) - server_error

NOTES

- This endpoint isn’t too important, but checks that a user is logged in (you will prob never use this on the Android end)
- If the “authorized” field is not present in the response, it is safe to assume that the user is not logged in.

EXAMPLE

Request: GET https://vptech-fitness.herokuapp.com/oauth/validate
Response:

```
{
      authorized: true,
}
```

Get Current User

ENDPOINT: /apis/users
HTTP REQUEST: GET
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- None

RESPONSE (200 OK)

- status VARCHAR (string) - “success” if it worked
- data OBJECT - consists of the fields below
  - id NUMBER - their ID in the database
  - username VARCHAR (string)
  - email VARCHAR (string)
  - name VARCHAR (string)
  - active BOOLEAN (true | false)
  - created TIMESTAMP

RESPONSE (500 ERROR)

- status VARCHAR (string)
- message OBJECT - The type of error

NOTES

- This endpoint is used to get account information about the currently logged in user (obtained from the access token). The most relevant fields for the front end are the username, email, name, and created.

EXAMPLE

Request: GET https://vptech-fitness.herokuapp.com/apis/users

Response:

```
{
      "status": "success",
      "data": {
            "id": 5,
            "username": "vontell",
            "email": "vontell@mit.edu",
            "name": "Aaron Vontell",
            "active": true,
            "created": "2017-07-29T22:23:02.858Z"
      }
}
```

Update Current User

ENDPOINT: /apis/users
HTTP REQUEST: PATCH
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- Same format at “Create User Info”, except all fields are optional AND there cannot be a password, username, id, or created field.

RESPONSE (200 OK)

- The same response as “Get Current Use”, with the updates created

RESPONSE (500 ERROR)

- status VARCHAR (string) - “failure”
- message VARCHAR (string) - What the error was

NOTES

- This will usually respond with a 500 error if the user does not exist, or a forbidden field is given (such as username, password, id, created).

EXAMPLE
Request: PATCH https://vptech-fitness.herokuapp.com/apis/users
Body:

```
{
      active: false
}
```

Response:

```
{
```

```
        status: "success",
        data: {
              id: 5,
              username: "vontell",
              email: "vontell@mit.edu",
              name: "Aaron Vontell",
              active: false,
              created: "2017-07-29T22:23:02.858Z"
        }
  }
```

User Info API

The following are endpoints for creating, updating, and accessing information about users, or basic fitness preferences information.

Create User Info

ENDPOINT: /apis/users/info
HTTP REQUEST: POST
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- age INTEGER
- weight DECIMAL
- height DECIMAL
- goal_weight DECIMAL
- difficulty INTEGER
- equipment INTEGER[]

RESPONSE (200 OK)

- status VARCHAR (string) - “success” if it worked
- message VARCHAR (string) - “User info added”

RESPONSE (500 ERROR)

- status VARCHAR (string) - “failure”
- message VARCHAR (string) - What the error was

NOTES

- The fields above are all optional; in other words, some of these may be undefined or null, and we should make sure that the frontend handles that
- The array of equipment is handled by wrapping the integers in curly braces, i.e. [1,2,3] = “{1,2,3}”.

EXAMPLE

Request: POST https://vptech-fitness.herokuapp.com/apis/users/info
Body:

```
{
      age: 21,
      weight: 160,
      height: 67,
      goal_weight: 130,
      difficulty: 3,
      equipment: {1,5,2,12}
}
```

Response:

```
{
      status: "success",
      message: "User info added"
}
```

Get User Info

ENDPOINT: /apis/users/info
HTTP REQUEST: GET
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- None

RESPONSE (200 OK)

- status VARCHAR (string) - “success” if it worked
- data OBJECT - The user info
    - id INTEGER - id of this info in the DB
    - username VARCHAR (string) - the user that this belong to
    - age INTEGER
    - weight VARCHAR (but represents a decimal)
    - height VARCHAR (but represents a decimal)
    - goal_weight VARCHAR (but represents a decimal)
    - difficulty INTEGER
    - equipment INTEGER[]

RESPONSE (500 ERROR)

- status VARCHAR (string) - “failure”
- message VARCHAR (string) - What the error was

NOTES

- This will usually respond with a 500 error if the user info does not exist, and will have a message such as “No user info found for this user”.

EXAMPLE

Request: GET https://vptech-fitness.herokuapp.com/apis/users/info

Response:

```
{
      "status: "success",
      "data: {
            id: 3,
            username: "vontell",
            age: 21,
            weight: "160",
            height: "67",
            goal_weight: "130",
            difficulty: 3,
            equipment: [
                  1,
```

```
                    5,
                    2,
                    12
                ]
            }
        }
```

Update User Info

ENDPOINT: /apis/users/info
HTTP REQUEST: PATCH
HEADERS:

- Authorization: “Bearer access_token” (where access_token was obtained from /oauth/token)

BODY

- Same format at “Create User Info”, except any field can be passed

RESPONSE (200 OK)

- The same response as “Get User Info”, with the updates created

RESPONSE (500 ERROR)

- status VARCHAR (string) - “failure”
- message VARCHAR (string) - What the error was

NOTES

- This will usually respond with a 500 error if the user info does not exist, and will have a message such as “No user info found for this user”.

EXAMPLE

Request: PATCH https://vptech-fitness.herokuapp.com/apis/users/info
Body:

```
{
    height: 45
}
```

Response:

```
      {
            "status: "success",
            "data: {
                  id: 3,
                  username: "vontell",
                  age: 21,
                  weight: "160",
                  height: "45",
                  goal_weight: "130",
                  difficulty: 3,
                  equipment: [
                        1,
                        5,
                        2,
                        12
                  ]
            }
      }
```