ID5: API Documentation

Overview

The BEAP Engine API provides programmatic access to our services, allowing users to upload, process, and predict physical activity data from wearable devices like Fitbit, Apple Watch, and Garmin. This API is designed for researchers and developers looking to automate data analysis within their applications.

Base URL

All API requests are made to the base URL: `https://beapengine.com/api`

Authentication

The BEAP Engine API uses session cookie id for authentication. To obtain a token, use the login endpoint with your user credentials. Include the token in the cookie header of your requests.

Endpoints

Delete User Data

DELETE `/delete-User-Data`

Deletes personal data

- **Headers:**
  - `Cookie: SESSION=SESSIONID`
- **Success Response:**
  - **Code:** 200 OK
- **Error Response:**
  - **Code:** 203 Non authorative information
    - **Content:** `{ success: false, message: Invalid session }`
  - **Code:** 404 Not found

- Services called
    - `UserController → UserService.get → userDao.get → UserService.deleteUserData → UserDao.deleteUserData`

Delete Account

DELETE `/delete-Profile`

Deletes account and personal data

- **Headers:**
    - `Cookie: SESSION=SESSIONID`
- **Success Response:**
    - **Code:** 200 OK
- **Error Response:**
    - **Code:** 203 Non authorative information
        - **Content:** `{ success: false, message: Invalid session }`
    - **Code:** 404 Not found
- Services called
    - `UserController → UserService.get → userDao.get → UserService.deleteUserAccount → UserDao.deleteUserAccount`

Change Password

POST `/password`

Change password details

- **Headers:**
    - `Cookie: SESSION=SESSIONID`
- **Success Response:**
    - **Code:** 200 OK
- **Error Response:**
    - **Code:** 203 Non authorative information

        - **Content:** `{ success: false, message: Invalid session }`
    - **Code:** 404 Not found
        - **Content:** `{ success: false, message: User not found }`
- Services called
    - `UserController → UserService.get → userDao.get → UserService.newUpdateUser → UserDao.newUpdate`

Get User Details

GET `/user`

Gets user details

- **Headers:**
    - `Cookie: SESSION=SESSIONID`
- **Success Response:**
    - **Code:** 200 OK
        - Content `{ firstName: XYZ, lastName: XYZ, userName: XYZ }`
- **Error Response:**
    - **Code:** 203 Non authorative information
        - **Content:** `{ success: false, message: Invalid session }`
    - **Code:** 404 Not found
        - **Content:** `{ success: false, message: User not found }`
- Services called
    - `UserController → UserService.get → userDao.get`

Signup

POST `/user`

Signs up a new user

- **Body Parameters:**
    - `username` (required): User's username.

    - `password` (required): User's password.
    - `firstName` (required): User's first name.
    - `lastName` (required): User's last name.
    - `accessGroup` (required): 4 for user and 5 for admin
    - `role` (required): 4 for admin and 5 for user
- **Success Response:**
    - **Code:** 200 OK
- **Error Response:**
    - **Code:** 400 Bad request
    - **Content:** `{ success: false, "message": "Invalid User information provided" }`
- Services called
    - `UserController → UserService.save → userDao.save`

Login

POST `/loginuser`

Authenticates the user and returns a bearer token.

- **Body Parameters:**
    - `username` (required): User's username.
    - `password` (required): User's password.
- **Success Response:**
    - **Code:** 200 OK
    - **Content:** `{ "token": "YOUR_ACCESS_TOKEN" }`
- **Error Response:**
    - **Code:** 401 Unauthorized
    - **Content:** `{ success: false, "message": "Invalid email or password." }`
- Services called

- `LoginUserController → UserService.loadByUsername → userDao.findByUsername`

Upload Data

POST `/rest/beapengine/{watchType}/upload`

Allows users to upload data files for processing.

- **Parameters:**
  - `file: the zipped file to be uploaded as multipart/form-data`
  - `watchType: the watch type to be uploaded`
- **Headers:**
  - `Cookie: SESSION=SESSIONID`
- **Success Response:**
  - **Code:** 200 OK
    - **Content:** `{ success: true, raw_data_id: XXXX }`
- **Error Response:**
  - **Code:** 400 Bad Request
    - **Content:** `{ success: false, message: Invalid watch type }`
  - **Code:** 406 Not Acceptable
    - **Content:** `{ success: false, message: One zip file is acceptable }`
  - **Code:** 415 Unsupported Media Type
    - **Content:** `{ success: false, message: Invalid file type }`
  - **Code:** 500 Server Error
    - **Content:** `{ success: false, message: XYZ }`
- **Services called**
  - `WatchController → WatchService.uploadAndPersist() → RawDataService.save() → RawDataDao.save()`

Process Data

GET `/rest/beapengine/{watchType}/process/{id}`

Initiates processing of the uploaded data file.

- **Parameters:**
  - `id: id of the file to be predicted which is the raw data id`
  - `watchType: the watch type to be processed`
- **Headers:**
  - `Cookie: SESSION=SESSIONID`
- **Path Parameters:**
  - `fileId`: The ID of the file to process.
- **Success Response:**
  - **Code:** 200 OK
  - **Content:** `{ success: true, processed_id: XXXX }`
- **Error Response:**
  - **Code:** 400 Bad Request
    - **Content:** `{ success: false, message: Invalid Watch type }`
  - **Code:** 500 Bad Request
    - **Content:** `{ success: false, message: XYZ }`
- **Services called**
  - `WatchController → WatchService.predictData() → RawDataService.get() → RawDataDao.get() → R-Repository/DataProcessor.R → ProcessedDataService.save() → ProcessedDataService.save()`

Predict Activity

GET `/rest/beapengine/{watchType}/predict/{id}/{predictionmodel}`

Returns predictions based on processed data.

- **Parameters:**
  - `id: id of the file to be predicted which is the processed data id`
  - `predictionmodel: can be svm, randomForest, rotationForest, and decissionTree`

    - `watchType: the watch type to be predicted`
- **Headers:**
    - `Cookie: SESSION=SESSIONID`
- **Success Response:**
    - **Code:** 200 OK
        - **Content:** `{ success: true, predicted_id : XXXX }`
- **Error Response:**
    - **Code:** 406 Not Acceptable
        - **Content:** `{ success: false, message: Invalid predictionModel }`
    - **Code:** 500 Server Error
        - **Content:** `{ success: false, message: XYZ }`
- **Services called**
    - `WatchController → WatchService.predictData() → ProcessedDataService.get() → ProcessedDataDao.get() → R-Repository/Predictor.R → PredictedDataService.save() → PredictedDataDao.save()`

Download Data

GET `/rest/beapengine/{watchType}/download_file/{id}/{type}`

Downloads types of processed files or predicted files

- **Parameters:**
    - `id: id of file to be downloaded`
    - `type: type of file to be downloaded`
    - `watchType: the watch type to be downloaded`
- **Headers:**
    - `Cookie: SESSION=SESSIONID`
- **Success Response:**
    - **Code:** 200 OK

        - **Content:** `{success: true, file: the_file}`

- **Error Response:**
    - **Code:** 404 Not Found
        - **Content:** `{success: false, message: Requested .zip file not found at the server}`

- **Services called**
    - `WatchController -> WatchService.download(id, type) -> ProcessedDataService.get()-> ProcessedDataDao.get() -> PredictedDataService.get() -> PredictedDataDao.get()`