swagger

Select a spec default

Api Documentation

1.0

[Base URL: localhost:8080/]

http://localhost:8080/v2/api-docs

Api Documentation

Terms of service

Apache 2.0

apple-watch-controller

Apple Watch Controller

GET /rest/beapengine/applewatch/download_file/{id}/{type} Apple Watch file download

Downloading the processed/predicted Apple Watch files

Parameters

Try it out

Name	Description
id * required string *(path)*	id of the file to be downloaded
type * required string *(path)*	type of file to be downloaded: process or predict

Responses

Response content type */*

Code	Description
200	*{success: true, file: the_file}*
401	*Unauthorized*
403	*Forbidden*
404	*{success: false, message: Requested .zip file not found at the server}*

Example Value Model

```
{
  "additionalProp1": {},
  "additionalProp2": {},
  "additionalProp3": {}
}
```

GET `/rest/beapengine/applewatch/list/{type}` Apple Watch data list

List the user's Apple Watch data.

Parameters

Try it out

Name	Description
type * required `string` *(path)*	type of the apple watch data to be listed: raw, processed, predicted

Responses

Response content type */*

Code	Description

200	*{success: true, predicted_id: XXXX}*
	
	Example Value Model
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: Invalid predictionModel}*
500	*{success: false, message: XYZ}*

GET `/rest/beapengine/applewatch/predict/{id}/{predictionmodel}` Apple Watch data prediction

Predicting the user's Apple Watch data, which has been processed before.

Parameters Try it out

Name	Description
id * required string *(path)*	id of the file to be predicted, which is the processed data id

Name	Description
predictionmodel * required string *(path)*	the machine learning model: svm, randomForest, rotationForest, decissionTree

Responses

Response content type */*

Code	Description

200	*{success: true, predicted_id: XXXX}*
	
	Example Value Model
	
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: Invalid predictionModel}*
500	*{success: false, message: XYZ}*

GET `/rest/beapengine/applewatch/process/{id}` Apple Watch data processing

Processing the user’s Apple Watch data, which has been uploaded before.

Parameters Try it out

Name	Description
id * required string *(path)*	id of the file to be processed, which is the raw data id

Responses Response content type */*

Code	Description

200	*{success: true, processed_id: XXXX}*
	
	Example Value Model
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
500	*{success: false, message: XYZ}*

POST `/rest/beapengine/applewatch/upload` Zip file upload

Uploading a zip file containing Apple Watch raw data and persisting it to database

Parameters Try it out

Name	Description

file * required
(blob)
	the zipped file to be uploaded as multipart/form-data, being processData: false, contentType: false in the ajax request.
	A sample ajax request is as follows:
	
	--------begin example-------
	var formData = new FormData();
	formData.append('fname', 'applewatch.zip');
	formData.append('data', blob);
	$.ajax({
	type: 'POST',
	url: '/rest/beapengine/applewatch/upload',
	data: formData,
	processData: false,
	contentType: false,
	success: function (data, status, xhr) {
	console.log('status: ' + status + ', data: ' + data);
	},
	error: function (jqXhr, textStatus, errorMessage) {
	var result = JSON.parse(jqXhr.responseText);
	console.log("error: " + result)
	}
	});
	--------end example-------

Responses

Response content type */*

Code	Description
200	*{success: true}, raw_data_id: XXXX}* **Example Value** Model { "additionalProp1": {}, "additionalProp2": {},

Code	Description

	"additionalProp3": {}
	}

201	*Created*
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: One zip file is acceptable}*
415	*{success: false, message: Invalid file type}*
500	*{success: false, message: XYZ}*

GET `/rest/beapengine/applewatch/uploadview` Apple Watch view

Returns the Apple Watch view page

Parameters Try it out

No parameters

Responses Response content type `*/*`

Code	Description
200	*OK*

Code	Description
	Example Value Model

```
{
  "empty": true,
  "model": {},
  "modelMap": {
    "additionalProp1": {},
    "additionalProp2": {},
    "additionalProp3": {}
  },
  "reference": true,
  "status": "100",
  "view": {
    "contentType": "string"
  },
  "viewName": "string"
}
```

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

fitbit-controller Fitbit Controller

GET `/rest/beapengine/fitbit/download_file/{id}/{type}` Fitbit file download

Downloading the processed/predicted Fitbit files

Parameters Try it out

Name	Description
id * required `string` *(path)*	id of the file to be downloaded

Name	Description
type * required `string` *(path)*	type of file to be downloaded: process or predict

Responses Response content type `*/*`

Code	Description

200	*{success: true, file: the_file}*
	
	Example Value Model
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*{success: false, message: Requested .zip file not found at the server}*

GET `/rest/beapengine/fitbit/list/{type}` Fitbit data list

List the user's Fitbit data.

Parameters Try it out

Name	Description
type * required `string`	type of the fitbit data to be listed: raw, processed, predicted

Name	Description
(path)	

Responses Response content type `*/*`

Code	Description
200	*{success: true, predicted_id: XXXX}*
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: Invalid predictionModel}*
500	*{success: false, message: XYZ}*

Example Value Model (for 200)

```
{
  "additionalProp1": {},
  "additionalProp2": {},
  "additionalProp3": {}
}
```

GET `/rest/beapengine/fitbit/predict/{id}/{predictionmodel}` Fitbit data prediction

Predicting the user's Fitbit data, which has been processed before.

Parameters Try it out

Name	Description
id * required string *(path)*	id of the file to be predicted, which is the processed data id
predictionmodel * required string *(path)*	the machine learning model: svm, randomForest, rotationForest, decissionTree

Responses **Response content type** `*/*`

Code	Description

200	*{success: true, predicted_id: XXXX}*
	
	Example Value Model
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: Invalid predictionModel}*
500	*{success: false, message: XYZ}*

GET `/rest/beapengine/fitbit/process/{id}` Fitbit data processing

Processing the user's Fitbit data, which has been uploaded before.

Parameters Try it out

Name	Description
id * required string *(path)*	id of the file to be processed, which is the raw data id

Responses Response content type `*/*`

Code	Description

200	*{success: true, processed_id: XXXX}*
	
	Example Value Model
	`{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
500	*{success: false, message: XYZ}*

POST `/rest/beapengine/fitbit/upload` Zip file upload

Uploading a zip file containing Fitbit raw data and persisting it to database

Parameters

Try it out

Name	Description
file * required *(blob)*	the zipped file to be uploaded as multipart/form-data, being processData: false, contentType: false in the ajax request. A sample ajax request is as follows:

```
--------begin example-------
var formData = new FormData();
formData.append('fname', 'fitbit.zip');
formData.append('data', blob);
$.ajax({
type: 'POST',
url: '/rest/beapengine/fitbit/upload',
data: formData,
processData: false,
contentType: false,
success: function (data, status, xhr) {
console.log('status: ' + status + ', data: ' + data);
},
error: function (jqXhr, textStatus, errorMessage) {
var result = JSON.parse(jqXhr.responseText);
console.log("error: " + result)
}
});
--------end example-------
```

Responses

Response content type */*

Code	Description
200	*{success: true}, raw_data_id: XXXX}*

Code	Description
	Example Value Model `{ "additionalProp1": {}, "additionalProp2": {}, "additionalProp3": {} }`
201	*Created*
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*
406	*{success: false, message: One zip file is acceptable}*
415	*{success: false, message: Invalid file type}*
500	*{success: false, message: XYZ}*

GET `/rest/beapengine/fitbit/uploadview` Fitbit view

Returns the Fitbit view page

Parameters Try it out

No parameters

Responses **Response content type** */*

Code	Description
200	*OK*

Example Value Model

```
{
  "empty": true,
  "model": {},
  "modelMap": {
    "additionalProp1": {},
    "additionalProp2": {},
    "additionalProp3": {}
  },
  "reference": true,
  "status": "100",
  "view": {
    "contentType": "string"
  },
  "viewName": "string"
}
```

401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

home-controller Home Controller

GET / handleRequest

Parameters Try it out

No parameters

Responses **Response content type** */*

Code	Description
200	*OK* **Example Value** Model

```
{
  "empty": true,
  "model": {},
  "modelMap": {
    "additionalProp1": {},
    "additionalProp2": {},
    "additionalProp3": {}
  },
  "reference": true,
  "status": "100",
  "view": {
    "contentType": "string"
  },
  "viewName": "string"
}
```

Code	Description
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

login-user-controller Login User Controller

POST `/loginuser` Login User

Logging in a user with username and password

Parameters Try it out

Name	Description
password * required `string` *(query)*	password

Name	Description
username * required string *(query)*	username

Responses

Response content type */*

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.762Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.762Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.762Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
          "hours": 0,
          "minutes": 0,
```

Code	Description
201	*Created*
401	*Unauthorized*
403	*Forbidden*

Code	Description
404	*Not Found*

GET **/logoutuser** Logout User

Logging out a user with username

Parameters Try it out

No parameters

Responses **Response content type** */*

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.762Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.762Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.762Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
```

Code	Description
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

```
            "hours": 0,
            "minutes": 0,
```

GET `/rest/beapengine/loginuser` List all LoginUsers

Returns a list of type LoginUserDto

Parameters

Try it out

No parameters

Responses

Response content type application/json

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.763Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.763Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.763Z"
      }
    ],
    "firstName": "string",
```

Code	Description
	```
	"id": 0,
	"lastName": "string",
	"password": "string",
	"rawData": [
	  {
	    "data": "string",
	    "dateTime": {
	      "date": 0,
	      "day": 0,
	      "hours": 0,
	```
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

POST `/rest/beapengine/loginuser` Add a new LoginUser

Saving an LoginUser and returning the saved LoginUser in an object of type LoginUserDto

Parameters

Try it out

Name	Description
loginUserDto * required *(body)*	loginUserDto

Example Value Model

```
{
  "date": "2024-03-18T00:41:20.631Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:20.631Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:20.631Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
```

Name	Description

```
        "rawData": [
          {
            "data": "string",
            "dateTime": {
              "date": 0,
              "day": 0,
              "hours": 0,
```

Parameter content type

application/json

Responses **Response content type** application/json

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.764Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.764Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.764Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
          "hours": 0,
          "minutes": 0,
```

201	*Created*

Code	Description
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

PUT `/rest/beapengine/loginuser` Update existing LoginUser

Updating an existing LoginUser and returning the updated LoginUser in an object of type LoginUserDto

Parameters

Try it out

Name	Description
loginUserDto * required *(body)*	loginUserDto

Example Value Model

```
{
  "date": "2024-03-18T00:41:19.820Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:19.820Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:19.820Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
          "hours": 0,
          "minutes": 0,
```

Parameter content type

Name	Description
	application/json

Responses Response content type application/json

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.765Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.765Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.765Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
          "hours": 0,
          "minutes": 0,
```

Code	Description
201	*Created*
401	*Unauthorized*
403	*Forbidden*

Code	Description
404	*Not Found*

GET `/rest/beapengine/loginuser/{id}` Find LoginUser by ID

Finding an LoginUser by input id and returning the result in an object of type LoginUserDto

Parameters

Try it out

Name	Description
id * required `string` *(path)*	id

Responses

Response content type application/json

Code	Description
200	*OK*

Example Value Model

```
{
  "date": "2024-03-18T00:41:28.766Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.766Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.766Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
```

Code	Description

```
      "password": "string",
      "rawData": [
        {
          "data": "string",
          "dateTime": {
            "date": 0,
            "day": 0,
            "hours": 0,
```

Code	Description
401	*Unauthorized*
403	*Forbidden*
404	*Not Found*

DELETE `/rest/beapengine/loginuser/{id}` Delete LoginUser by ID

Deleting an LoginUser with ID = id

Parameters

Try it out

Name	Description
id * required string *(path)*	id

Responses

Response content type application/json

Code	Description
200	*OK*

Example Value Model

Code	Description

```
{
  "date": "2024-03-18T00:41:28.767Z",
  "id": 0,
  "passwordToken": "string",
  "user": {
    "accessGroup": [
      {
        "accessGroupName": "string",
        "createdDate": "2024-03-18T00:41:28.767Z",
        "description": "string",
        "id": 0,
        "lastModifiedDate": "2024-03-18T00:41:28.767Z"
      }
    ],
    "firstName": "string",
    "id": 0,
    "lastName": "string",
    "password": "string",
    "rawData": [
      {
        "data": "string",
        "dateTime": {
          "date": 0,
          "day": 0,
          "hours": 0,
          "minutes": 0,
```

204	*No Content*
401	*Unauthorized*
403	*Forbidden*

Models

```
AccessGroupDto  {
    accessGroupName     string
    createdDate         string($date-time)
    description         string
    id                  integer($int64)
    lastModifiedDate    string($date-time)
}
```

```
LoginUserDto {
   date                 string($date-time)
   id                   integer($int64)
   passwordToken        string
   user                 UserDto {...}
}
```

```
ModelAndView {
   empty                boolean
   model                   {...}
   modelMap                {...}
   reference            boolean
   status               string
                        Enum:
                           Array [ 64 ]
   view                 View {...}
   viewName             string
}
```

```
RawDataDto {
   data                 string($byte)
   dateTime             Timestamp {...}
   id                   integer($int64)
   processedDataID      integer($int64)
   type                 string
                        Enum:
                           Array [ 2 ]
}
```

```
RoleDto {
   description          string
   id                   integer($int64)
   roleName             string
}
```

```
Timestamp {
    date              integer($int32)
    day               integer($int32)
    hours             integer($int32)
    minutes           integer($int32)
    month             integer($int32)
    nanos             integer($int32)
    seconds           integer($int32)
    time              integer($int64)
    timezoneOffset    integer($int32)
    year              integer($int32)
}
```

```
UserDto {
    accessGroup
                          [...]
    firstName         string
    id                integer($int64)
    lastName          string
    password          string
    rawData
                          [...]
    role
                          [...]
    username          string
}
```

```
View {
    contentType       string
}
```