REST API

Authentication

Get session

Retrieves the session of the user.

Request

```
GET /api/me/session
```

Response, session already exists:

```
HTTP/1.1 200 Ok
Access-Control-Allow-Origin: <ORIGIN>
Access-Control-Allow-Credentials: true
Set-Cookie: keystone.uid=<TOKEN>; Path=/; HttpOnly
Set-Cookie: keystone.sid=<TOKEN>; Path=/; HttpOnly
```

```
{
    "id": "53a984cca87b4b7d57a99858",
    "email": "john.doe@example.com",
    "name": {
        "first": "John",
        "last": "Doe"
    },
    "_csrf": "<CSFR TOKEN>",
}
```

Response, session doesn't exist:

```
HTTP/1.1 401 Unauthorized
```

```
{
    "code": "401",
    "status": "401",
    "name": "Http401Error",
    "message": "Unauthorized",
    "reason": {
          "name"    : "AuthenticationError",
          "message" : "No session exists."
    }
}
```

Signin

Request

(Re-)creates a session.

```
POST /api/me/session
```

```
{
    "email": "john.doe@example.com",
    "password": "42"
}
```

Response: Success

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: <ORIGIN>
Access-Control-Allow-Credentials: true
Set-Cookie: keystone.uid=<TOKEN>; Path=/; HttpOnly
Set-Cookie: keystone.sid=<TOKEN>; Path=/; HttpOnly
```

```
{
    "id": "53a984cca87b4b7d57a99858",
    "email": "john.doe@example.com",
    "name": {
        "first": "John",
        "last": "Doe"
    },
    "_csrf": "<CSRF TOKEN>"
}
```

Response: Failure

```
HTTP/1.1 401 Unauthorized
```

```
{
    "code": "401",
    "status": "401",
    "name": "Http401Error",
    "message": "Unauthorized",
    "reason": {
          "name"    : "AuthenticationError",
          "message" : "Bad credentials."
    }
}
```

Signout

Request

Destroys the current session.

```
DELETE /api/me/session
```

Response

```
HTTP/1.1 204 No Content
```

Logged in user

Retrieve

Request

```
GET /api/me/account
```

Response

```
HTTP/1.1 200 OK
```

```
{
  "id": "53a984cca87b4b7d57a99858",
  "email": "john.doe@example.com",
  "name": {
    "first": "John",
    "last": "Doe"
  }
}
```

Full Update

Request

```
PUT /api/me/account
```

```
{
  "email": "changedemail@example.com"
  "name": {
    "first": "John",
    "last": "Doe"
  },
```

```
  "password": "HolyShizzle!"
  "password_confirm": "HolyShizzle!"
  "_csrf": "<CSRF TOKEN>"
}
```

Response

```
HTTP/1.1 200 OK
```

```
{
  "id": "53a984cca87b4b7d57a99858",
  "email": "changedemail@example.com",
  "name": {
    "first": "John",
    "last": "Doe"
  }
}
```

Partial Update

Request

```
PATCH /api/me/account
```

```
{
  "email": "changedemail@example.com",
  "_csrf": "<CSRF TOKEN>"
}
```

N.B.: though `password` is never returned as a user object, it can be updated too. It does however need an additional value: `password_confirm`.

E.g.:

```
{
    "password": "HolyShizzle!"
    "password_confirm": "HolyShizzle!"
}
```

Response

```
HTTP/1.1 200 OK
```

```
{
  "id": "53a984cca87b4b7d57a99858",
```

```
  "email": "changedemail@example.com",
  "name": {
    "first": "John",
    "last": "Doe"
  }
}
```

Comparisons

Retrieve current (active) comparison for the logged in user

Request

```
GET /api/me/comparison
```

Response: found

```
HTTP/1.1 200 OK
```

```
{

}
```

Response: not found

```
HTTP/1.1 204 OK
```

Create an (active) comparison for the logged in user

Request

```
POST /api/me/comparison
```

```
{
  "assessment" : "53a984cca87b4b7d57a99858"
}
```

Response

```
HTTP/1.1 200 OK
```

```
{
```

```
}
```

Errors

All error objects have a similar (base) structure:

```
{
  "code": "404",
  "status": "404",
  "name": "Http404Error",
  "message": "Not Found"
}
```

(Quotes taken from wikipedia) Depending on the requested operation and largely based on the guidelines as laid out in the HTTP API Design Guide, following status codes are returned by the API methods:

401 Unauthorized

> Authentication is required and has failed or has not yet been provided.

403 Forbidden

> The request was a valid request, but the server is refusing to respond to it. Unlike a 401 Unauthorized response, authenticating will make no difference.

These errors include failure to comply due to a missing parameter.

404 Not Found

> The requested resource could not be found but may be available again in the future. Subsequent requests by the client are permissible.

405 Method Not Allowed

> A request was made of a resource using a request method not supported by that resource; for example, using GET on a form which requires data to be presented via POST, or using PUT on a read-only resource.

422 Unprocessable Entity

> The request was well-formed but was unable to be followed due to semantic errors.

These errors occur when an operation on an entity fails, not due to malformed syntax, but because of a missing operator or operand, incorrect data type, et cetera. 422 errors provide an extra field `reason` with a machine and human readable explanation on what went wrong.

E.g.:

```
{
  "code": "422",
  "status": "422",
  "name": "Http422Error",
  "message": "Unprocessable Entity",
  "reason": {
    "message": "Validation failed",
    "name": "ValidationError",
    "errors": {
      "password": {
        "name": "ValidatorError",
        "path": "password",
        "message": "Passwords must match",
        "type": "required"
      }
    }
  }
}
```

500 Internal Server Error

> A generic error message, given when an unexpected condition was encountered and no more specific message is suitable.