Vending App

Overview

OpenAPI Doc for the vending machine app, automatically generated using fastify-swagger

Terms Of Service

Security

Type	Name	Scopes
apiKey	session_cookie	

Paths

POST `/api/v1.0/age_verification/{mode}/{id}` Perform age verification on an image

This is the route primarily used by our frontend

Parameters

Type	Name	Description	Schema
path	**mode** *required*		
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/client_sse/{age}` Client SSE

Parameters

Type	Name	Description	Schema
path	**age** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/contact` Send a bug report to developers

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/frontend_sse` Frontend SSE

Currently not implemented, will only return a sample event

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/migrate` Apply migrations

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/admin/login` Confirm admin login

Parameters

Type	Name	Description	Schema
query	**token** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/admin/login` Request admin login

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/admin/devices/request_log` Request that a device uploads its serial log

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/admin/users/` Read Users

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/admin/users/` Create a new User

Responses

Code	Description	Links
200	Default Response	No Links

DELETE `/api/v1.0/admin/users/` Delete Users

Responses

Code	Description	Links
200	Default Response	No Links

PATCH `/api/v1.0/admin/users/` Update User

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/admin/users/impersonate_user` Impersonate a User

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/admin/users/{id}` Read Users

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/admin/users/search` Search Users

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/auth/forward` Forward Auth

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/auth/has_session` Get session state

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/auth/login` Login Step 1

Login using Email and Password for Users and SubUsers

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/auth/login_two_fa` Login Step 2

Second login factor, using TOTP

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/auth/logout` Logout

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/branding/` Read All

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/branding/` Create

Responses

Code	Description	Links
200	Default Response	No Links

DELETE `/api/v1.0/branding/` Delete

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/branding/{id}/get_logo` Get logo

ID can be a device ID, user ID or session ID

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/branding/{id}` Read

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

PUT `/api/v1.0/branding/{id}` Update

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/branding/{id}/upload_logo` Upload Logo

Body must be a multipart upload image file

It will be resized to 1:1 ratio

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/client/upload_log` Upload Log

For uploading debugging logs of the MDB bus, log is automatically sent to Chloe and Michael

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/devices/` Read All

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/devices/` Create

Responses

Code	Description	Links
200	Default Response	No Links

DELETE `/api/v1.0/devices/` Delete

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/devices/{id}` Read

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

PUT `/api/v1.0/devices/{id}` Update

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/devices/state/{id}` Get state

Doesn’t require log-in, for our website to test if a device is online and can do age verification

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/roles/` Read All

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/roles/` Create

See read all permissions for a list of possible values

Responses

Code	Description	Links
200	Default Response	No Links

DELETE `/api/v1.0/roles/` Delete

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/roles/all_permissions` Read All possible Permissions

Returns two arrays, one specifying all permissions that optionally can have Object IDs at the end (seperated with a ":") and one where that is not allowed

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/roles/{id}` Read

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

PUT `/api/v1.0/roles/{id}` Update

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/subusers/` Read All

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/subusers/` Create

After creation the new subuser will receive an E-Mail allowing them to set a password

Responses

Code	Description	Links
200	Default Response	No Links

DELETE `/api/v1.0/subusers/` Delete

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/subusers/{id}` Read

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

PUT `/api/v1.0/subusers/{id}` Update

Parameters

Type	Name	Description	Schema
path	**id** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/user/change_email` Confirm email change

Parameters

Type	Name	Description	Schema
query	**token** *required*		

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/change_email` Request email change

Will automatically delete preexisting requests if any exist for this account, leave new_email undefined to try to resend an existing request

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/confirm_email` Confirm email and set password

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/user/` Read current user

Responses

Code	Description	Links
200	Default Response	No Links

PUT `/api/v1.0/user/` Update

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/request_password_reset` Request password reset

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/reset_password` Use token to reset password

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/two_fa/setup` Setup 2FA

Can also be used to redo the setup

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/user/two_fa/confirm_setup` Confirm 2FA setup

Enter a code to confirm that the user has correctly saved the 2FA info

Responses

Code	Description	Links
200	Default Response	No Links

POST `/api/v1.0/verification/image` Perform age verification on an image

Takes an image as either a multipart file upload or string

Then does OCR and if everything was successful returns the age

If there was an error it will return a reason for the failure

Responses

Code	Description	Links
200	Default Response	No Links

GET `/api/v1.0/verification/session` Start a verification session

```
Starts a new session to verify a user's age via the IXATRIA website by opening a SSE
socket
```

```
Immediately after connecting a 'qr_code' event will be sent with a url in the data,
that the vending machine should show
```

```
Then it will send 'result' events whenever a user is attempting verification, the data
follows the same format as the image verification route
```

```
The 'result' event may be repeated often if the user is using video to verify
```

Responses

Code	Description	Links
200	Default Response	No Links

Components