

---

**beh.cam**

HTTP REST API Dokumentation

beh.digital GmbH

**beh.**  
digital

2022-08-11

## Inhaltsverzeichnis

<b>Vorwort</b>	<b>3</b>
Einführung . . . . .	3
Copyright . . . . .	4
Kontakt & Support . . . . .	4
Versionshistorie . . . . .	4
<b>HTTP REST Schnittstelle</b>	<b>5</b>
Spezifische Datentypen . . . . .	5
Tasktyp Datentyp . . . . .	5
<b>Task-spezifische Endpunkte</b>	<b>5</b>
Get Tasks . . . . .	5
Add Task . . . . .	7
Task Import (zip) . . . . .	7
Available Task Templates . . . . .	7
Get Task . . . . .	8
Export Task (zip) . . . . .	8
Delete Task . . . . .	9
Update Task . . . . .	9
Patch Task . . . . .	9
Task Image (first sample) . . . . .	9
Add Class to Task . . . . .	10
Delete Class from Task . . . . .	10
<b>Sample-spezifische Endpunkte</b>	<b>10</b>
Get Samples . . . . .	10
Add Sample . . . . .	11
Capture Image as new Sample . . . . .	12
Get Sample . . . . .	12
Delete Sample . . . . .	13
Patch Sample . . . . .	13
Get Image of Sample . . . . .	13
<b>Model-spezifische Endpunkte</b>	<b>13</b>
Inferenz-Ergebnis Struktur . . . . .	13
Get Models . . . . .	14
Add Model . . . . .	17

Import Model (zip) . . . . .	17
Get Model . . . . .	17
Patch Model . . . . .	19
Delete Model . . . . .	19
Infer/Execute Model (single image) . . . . .	20
Infer/Execute Model (stream) . . . . .	21
Export Model (as zip) . . . . .	22
Get File of Model . . . . .	22
Update File of Model . . . . .	22
<b>Video-spezifische Endpunkte</b>	<b>22</b>
H264-Stream (live) . . . . .	22
Video Einstellungen . . . . .	23
Video Einstellungen modifizieren . . . . .	23
Bild aufnehmen . . . . .	23
<b>Inferenz-spezifische Endpunkte</b>	<b>23</b>
<b>Einstellungs-spezifische Endpunkte</b>	<b>24</b>
GET Netzwerk-Einstellungen . . . . .	24
ADD IP-Adresse zur Netzwerk-Schnittstelle . . . . .	24
DELETE IP-Adresse von Netzwerk-Schnittstelle . . . . .	24
SW-Update der Kamera . . . . .	24
GET Autostart Modelle . . . . .	24
Füge Autostart Modell hinzu . . . . .	25
DELETE Autostart Model . . . . .	25
GET Hardware Status . . . . .	25
<b>Status-spezifische Endpunkte</b>	<b>25</b>
GET Service Status . . . . .	25
RPC Service . . . . .	25

## Vorwort

### Einführung

Die beh.digital GmbH hat dieses Handbuch gewissenhaft erstellt. Dennoch kann keine Gewähr für den Inhalt, die Vollständigkeit und die Qualität der Informationen in dessen erfolgen. In regelmäßigen

Abständen und bei Updates wird dieses Handbuch aktualisiert und überprüft. Bei fehlenden Informationen oder Problemen mit unserem Produkt wenden sie sich bitte an ihren Ansprechpartner oder schreiben sie uns eine E-Mail an: [support@beh.digital](mailto:support@beh.digital).

Auch beim Befolgen der in diesem Handbuch enthaltenden Bedienungsanleitung können wir nicht garantieren, dass es nicht zu Problem der Hardware und Software kommen kann. Des Weiteren können wir mit dem Kauf der Kamera nicht versprechen, dass das gewünschte Ergebnis erzielt wird. Soweit gesetzlich zulässig, ist die Haftung für direkte Schäden, Folgeschäden und Schäden Dritter, die sich aus dem Kauf dieses Produkts ergeben, ausgeschlossen. Die Haftung beschränkt sich auf den im Kaufvertrag angegebenen Produktpreis. Alle Verpflichtungen und Gewährleistungsregelungen sind ausschließlich in dem Kaufvertrag enthalten. Demnach stellt dieses Handbuch weder eine Erweiterung noch eine Einschränkung dar. Durch den Kauf wird dem Käufer das Recht zur Nutzung der Software eingeräumt.

## Copyright

© Alle Rechte dieses Handbuches sind der beh.digital GmbH vorbehalten. Kein Teil dieses Handbuchs darf ohne schriftliche Genehmigung der beh.digital GmbH verwendet werden. Dies schließt die Übersetzung in eine andere Sprache ein.

## Kontakt & Support

Sie erreichen uns unter

- 1 [beh.digital GmbH](#)
- 2 [Hespenkamp 1](#)
- 3 [DE-30419 Hannover](#)
- 4 [+49 \(0\)511 10582557](#)
- 5 [info@beh.digital](mailto:info@beh.digital)
- 6 <https://beh.digital>

## Versionshistorie

Version	Datum	Verantwortlicher	Bemerkung
1.0.0	11.08.2022	Behmann	Initiale Veröffentlichung

## HTTP REST Schnittstelle

Die beh.cam stellt eine HTTP REST-basierte API zur Verfügung. Diese wird ebenfalls von der integrierten Weboberfläche verwendet. Die Schnittstelle verfügt über keine internen Zustände, die die Ausgabe einer Anfrage beeinflusst.

- Etablierte Englisch-sprachige Fachbegriffe wurden in der nachfolgenden, Endpunkt-spezifischen Dokumentation übernommen und nicht ins Deutsche übersetzt.
- Exemplarische Rückgaben der Schnittstelle wurden zur besseren Lesbarkeit mit Zeilenumbrüchen dargestellt, die in der rohen Rückgabe nicht enthalten sind.
- Falls eine Header Regexp angegeben ist, muss diese erfüllt sein, um eine Ausgabe zu erhalten. Anderenfalls wird der Fehlercode 404 (nicht gefunden) zurückgegeben.

## Spezifische Datentypen

### Tasktyp Datentyp

Der *type* bei der Rückgabe eines Task kennzeichnet das verwendete Template:

<i>type</i>	Task-Type	Beschreibung
0	Klassifikation	Klassifikation eines Bild(ausschnitts) in eine trainierte Klasse
1	Detektion	Erkennung (Lokalisierung und Klassifikation) mehrerer Objekte innerhalb eines Bildes
2	Multi Task	Ausführung mehrerer Klassifikations-Modelle an festen Positionen
3	<i>reserviert</i>	<i>reserviert für zukünftige Anomalie-Erkennung</i>
4	Custom Task	Ausführung von Bildverarbeitung auf der Kamera

## Task-spezifische Endpunkte

### Get Tasks

Gibt die auf der Kamera gespeicherten Tasks / Aufgaben als JSON-Array zurück.

Endpunkte: /api/tasks

Methoden: GET

Headers Regexp: {"Accept": "application/json"}

Exemplarische Rückgabe:

```
1  [
2    {
3      "id": "c9on9kjp8j5s5onmtbgg",
4      "name": "Zahnraderkennung",
5      "type": 0,
6      "classes": [
7        {
8          "id": 2,
9          "name": "Belegt",
10         "profinetID": 1,
11         "gpio0": 1,
12         "gpio1": 0
13       },
14       {
15         "id": 3,
16         "name": "Leer",
17         "profinetID": 0,
18         "gpio0": 0,
19         "gpio1": 1
20       }
21     ],
22     "createdAt": "2022-05-03T20:30:42.373304+02:00",
23     "deletedAt": "0001-01-01T00:00:00Z"
24   },
25   {
26     "id": "c8kugu75upvevhdund4g",
27     "name": "Leerer Task",
28     "type": 0,
29     "createdAt": "2022-03-10T13:00:56.877834+01:00",
30     "deletedAt": "0001-01-01T00:00:00Z"
31   }
32 ]
```

Query Parameter: - **type**: auf einen oder mehrere Task-Typen filtern. Standard: Alle Tasktypen werden ausgegeben - **deleted**: wenn 1, werden bereits gelöschte Tasks zurückgegeben. Standard: Ausgabe nur noch nicht gelöschter Tasks - **limit**: maximale Anzahl an Tasks in der Ausgabe. Standard: 30 -

`offset`: Start der begrenzten Ausgabe an Tasks. Standard: 0

## Add Task

Hinzufügen eines Tasks.

Endpunkte: `/api/tasks`

Methoden: `POST`

Headers Regexp: `{"Accept": "application/json"}`

*Dokumentation folgt*

## Task Import (zip)

Importiert einen Task aus einem ZIP. Dieses kann beispielsweise über `GET Task (as zip)` (auf einem anderen Gerät) exportiert worden sein.

Endpunkte: `/api/tasks/import`

Methoden: `POST`

Headers Regexp: `{"Content-Type": "application/zip"}`

## Available Task Templates

Gibt die zur Verfügung stehenden Templates der Kamera zurück. Array der unterstützten Tasktypes.

Endpunkte: `/api/tasks/templates`

Methoden: `GET`

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 [
2   0,
3   2,
4   4
5 ]
```

## Get Task

Gibt die Eigenschaften eines Tasks zurück. Format identisch zu einem Element der Rückgabe aus [GET Tasks](#).

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}`

Methoden: GET

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 {
2   "id": "c9on9kjp8j5s5onmtbgg",
3   "name": "Zahnraderkennung",
4   "type": 0,
5   "classes": [
6     {
7       "id": 2,
8       "name": "Belegt",
9       "profinetID": 1,
10      "gpio0": 1,
11      "gpio1": 0
12    },
13    {
14      "id": 3,
15      "name": "Leer",
16      "profinetID": 0,
17      "gpio0": 0,
18      "gpio1": 0
19    }
20  ],
21  "createdAt": "2022-05-03T20:30:42.373304+02:00",
22  "deletedAt": "0001-01-01T00:00:00Z"
23 }
```

## Export Task (zip)

Exportiert einen Task als Zip-Datei, beispielsweise als Backup, oder zur Vervielfältigung des Tasks auf mehrere Endgeräte.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}`, `/api/tasks/{taskID:[0-9a-v]{20}}/task.zip`



Methoden: [GET](#)

Headers Regexp: `{"Accept": "application/zip"}, {}`

Exemplarische Rückgabe: ZIP-Datei des Tasks, bestehend aus Bildern, Modellen und gespeicherten Samples sowie Informationen zum Task.

## Delete Task

Löscht einen Task. Führt zunächst ein Soft-Delete (setzen des Feldes `deletedAt`) aus. Falls bereits Soft-Deleted, wird der Task hart gelöscht.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}`

Methoden: [DELETE](#)

Rückgaben:

Code	Status	Bedeutung
204	No Content	Erfolgreich gelöscht
400	Bad Request	Task-ID unbekannt oder anderer Fehler

## Update Task

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}`

Methoden: [PUT](#)

*Dokumentation folgt*

## Patch Task

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}`

Methoden: [PATCH](#)

*Dokumentation folgt*

## Task Image (first sample)

Gibt das Bild des ersten Samples aus einem Task zurück.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/image`

Methoden: `GET`

Ausgabe: `PNG`

### Add Class to Task

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/classes`

Methoden: `POST`

*Dokumentation folgt*

### Delete Class from Task

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/classes/{classID:[0-9]}`

Methoden: `DELETE`

*Dokumentation folgt*

## Sample-spezifische Endpunkte

### Get Samples

Gibt die Samples eines Tasks zurück.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples`

Methoden: `GET`

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 [
2   {
3     "id": "brk8grf5upv9q48s522g",
4     "taskID": "brk8d875upv9q48s51og",
5     "image": "/api/tasks/brk8d875upv9q48s51og/samples/
        brk8grf5upv9q48s522g/image",
6     "crop": {
7       "x": 0,
```

```
8         "y": 0,
9         "w": 1,
10        "h": 1
11    },
12    "validation": false,
13    "createdAt": "2020-06-16T10:53:01.036+02:00",
14    "deletedAt": "0001-01-01T00:00:00Z",
15    "classID": 1
16  },
17  {
18    "id": "brk8dvv5upv9q48s51sg",
19    "taskID": "brk8d875upv9q48s51og",
20    "image": "/api/tasks/brk8d875upv9q48s51og/samples/
21             brk8dvv5upv9q48s51sg/image",
22    "crop": {
23      "x": 0,
24      "y": 0,
25      "w": 1,
26      "h": 1
27    },
28    "validation": true,
29    "createdAt": "2020-06-16T10:46:54.924+02:00",
30    "deletedAt": "0001-01-01T00:00:00Z",
31    "classID": 2
32  },
33  ]
```

Query Parameter: - `class_id`: auf einen oder mehrere Klassen-IDs filtern. Ausschließlich bei Klassifikations-Modellen Standard: Alle Klassen-IDs werden berücksichtigt - `validation`: wenn gesetzt, werden nur Bilder mit `validation true` (1) oder `false` (0) zurückgegeben. Standard: 0 und 1 - `deleted`: wenn 1, werden bereits gelöschte Samples zurückgegeben. Standard: Ausgabe nur noch nicht gelöschter Samples - `limit`: maximale Anzahl an Tasks in der Ausgabe. Standard: 30 - `offset`: Start der begrenzten Ausgabe an Tasks. Standard: 0 - `random`: Zahl zwischen 0 und 1, mit welcher ein Seed der Sortierung vorgegeben werden kann. Standard: Sortierung nach ID

## Add Sample

Fügt ein Sample dem entsprechenden Task hinzu.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples`

Methoden: [POST](#)

*Dokumentation folgt*

### Capture Image as new Sample

Nimmt ein Bild auf und fügt dieses als neues (unklassifiziertes) Sample dem entsprechenden Task hinzu. Gibt dieses Sample zurück.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/capture`

Methoden: [GET](#)

Headers Regexp: `{"Content-Type": "application/json"}`

*Dokumentation folgt*

### Get Sample

Gibt das Sample mit entsprechender ID zurück.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples/{sampleID:[0-9a-v]{20}}`

Methoden: [GET](#)

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 {
2   "id": "c9s1qbf5upvbepg4aicg",
3   "taskID": "c9s1f075upva1tta4b40",
4   "type": 4,
5   "image": "/api/tasks/c9s1f075upva1tta4b40/samples/
6     c9s1qbf5upvbepg4aicg/image",
7   "crop": {
8     "x": 0,
9     "y": 0,
10    "w": 1,
11    "h": 1
12  },
13  "validation": false,
14  "createdAt": "2022-05-08T21:42:37.316948+02:00",
15  "deletedAt": "0001-01-01T00:00:00Z"
}
```

## Delete Sample

Löscht das Sample aus dem Task. Führt zunächst ein Soft-Delete (setzen des Feldes `deletedAt`) aus. Falls bereits Soft-Deleted, wird das Sample hart gelöscht.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples/{sampleID:[0-9a-v]{20}}`

Methoden: `DELETE`

Rückgaben:

Code	Status	Bedeutung
204	No Content	Erfolgreich gelöscht
400	Bad Request	Task-ID unbekannt oder anderer Fehler

## Patch Sample

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples/{sampleID:[0-9a-v]{20}|all}`

Methoden: `PATCH`

*Dokumentation folgt*

## Get Image of Sample

Gibt das Bild des Samples zurück.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/samples/{sampleID:[0-9a-v]{20}}/image`

Methoden: `GET`

Rückgabe: PNG

## Model-spezifische Endpunkte

### Inferenz-Ergebnis Struktur

Beispielhafte Ausgabe

```

1 {
2   "modelID": "c0mh5hv5upvdn1dnhs10",
3   "type": 1,
4   "roi": {
5     "x": 0,
6     "y": 0,
7     "w": 1,
8     "h": 1
9   },
10  "results": [
11    {
12      "classID": 1,
13      "mappedClassID": 0,
14      "className": "Objekt A",
15      "confidence": 0.3515625,
16      "bbox": {
17        "x": 0.1083346,
18        "y": 0.25184333,
19        "w": 0.73278534,
20        "h": 0.7098198
21      }
22    },
23    {
24      ...
25    }
26  ]
27 }

```

Eintrag	Datentyp	Beschreibung
modelID	string	ID des ausgeführten Modells
type	Tasktyp	Art des ausgeführten Modells
roi	Rectangle	Bildausschnitt, der betrachtet wurde
results	Inferenz-Ergebnis	Klassifikationsergebnis / Detektierte Objekte / Ausgabe

## Get Models

Endpunkte: [/api/tasks/{taskID:\[0-9a-v\]{20}}/models](/api/tasks/{taskID:[0-9a-v]{20}}/models)

Methoden: GET

Headers Regexp: {"Content-Type": "application/json"}

Exemplarische Rückgabe:

```
1  [
2    {
3      "id": "c9p500bp8j5sg3u9mu50",
4      "type": 0,
5      "taskID": "c9on9kjp8j5s5onmtbgg",
6      "name": "Unnamed Model",
7      "description": "No description",
8      "createdAt": "2022-05-04T09:08:17.716+02:00",
9      "deletedAt": "0001-01-01T00:00:00Z",
10     "augmentationConfig": {
11       "illuminationLower": 2,
12       "illuminationLowerStep": 1,
13       "illuminationUpper": 2,
14       "illuminationUpperStep": 1,
15       "grayscale": false,
16       "flipHorizontally": true,
17       "flipVertically": true,
18       "flipBoth": true,
19       "translationLeft": 0.1,
20       "translationLeftStep": 0.1,
21       "translationRight": 0.1,
22       "translationRightStep": 0.1,
23       "translationTop": 0.1,
24       "translationTopStep": 0.1,
25       "translationBottom": 0.1,
26       "translationBottomStep": 0.1,
27       "rotation": 0,
28       "rotationStep": 0,
29       "zoomOut": 1.2,
30       "zoomOutStep": 0.2,
31       "zoomIn": 1.2,
32       "zoomInStep": 0.2
33     },
34     "trainConfig": {
35       "classes": [
36         {
37           "id": 2,
38           "name": "Belegt",
```

```
39         "profinetID": 1,
40         "gpio0": 1,
41         "gpio1": 0
42     },
43     {
44         "id": 3,
45         "name": "Leer",
46         "profinetID": 0,
47         "gpio0": 0,
48         "gpio1": 1
49     }
50 ],
51 "name": "mobilenet_v2",
52 "inputSize": 224,
53 "depthMultiplier": 0.35,
54 "debug": false,
55 "hiddenUnits": 0,
56 "learningRate": 0.005,
57 "epochs": 100,
58 "batchSize": 100,
59 "validationSplit": -0.1,
60 "batchLogs": "[{"batch":0,"size":39,"loss
        \":3.109633445739746,\"acc\":0.07692307978868484},{...}]
        ",
61 "epochLogs": "[{"val_loss":2.3543283939361572,\"val_acc
        \":0.25641027092933655,\"loss\":3.109633445739746,\"acc
        \":0.07692307978868484},{...}]"
62 },
63 "inferenceConfig": {
64     "roi": {
65         "x": 0,
66         "y": 0,
67         "w": 1,
68         "h": 1
69     },
70     "gpio0func": 0,
71     "gpio1func": 0
72 }
73 },
74 {
75     ...
76 }
77 ]
```



## Add Model

Fügt ein Model einem Task hinzu.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/models`

Methoden: `POST`

Headers Regexp: `{"Content-Type": "multipart/form-data"}`

*Dokumentation folgt*

## Import Model (zip)

Importiert ein Model aus einer ZIP-Datei in einen Task.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/models/import`

Methoden: `POST`

Headers Regexp: `{"Content-Type": "application/zip"}`

*Dokumentation folgt*

## Get Model

Gibt das Model mit angefragter Model-ID zurück. Die Task-ID kann entfallen.

Endpunkte: `/api/models/{modelID:[0-9a-v]{20}}`, `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}`

Methoden: `GET`

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 {
2   "id": "c9p500bp8j5sg3u9mu50",
3   "type": 0,
4   "taskID": "c9on9kjp8j5s5onmtbgg",
5   "name": "Unnamed Model",
6   "description": "No description",
7   "createdAt": "2022-05-04T09:08:17.716+02:00",
```

```
8     "deletedAt": "0001-01-01T00:00:00Z",
9     "augmentationConfig": {
10         "illuminationLower": 2,
11         "illuminationLowerStep": 1,
12         "illuminationUpper": 2,
13         "illuminationUpperStep": 1,
14         "grayscale": false,
15         "flipHorizontally": true,
16         "flipVertically": true,
17         "flipBoth": true,
18         "translationLeft": 0.1,
19         "translationLeftStep": 0.1,
20         "translationRight": 0.1,
21         "translationRightStep": 0.1,
22         "translationTop": 0.1,
23         "translationTopStep": 0.1,
24         "translationBottom": 0.1,
25         "translationBottomStep": 0.1,
26         "rotation": 0,
27         "rotationStep": 0,
28         "zoomOut": 1.2,
29         "zoomOutStep": 0.2,
30         "zoomIn": 1.2,
31         "zoomInStep": 0.2
32     },
33     "trainConfig": {
34         "classes": [
35             {
36                 "id": 2,
37                 "name": "Belegt",
38                 "profinetID": 1,
39                 "gpio0": 1,
40                 "gpio1": 0
41             },
42             {
43                 "id": 3,
44                 "name": "Leer",
45                 "profinetID": 0,
46                 "gpio0": 0,
47                 "gpio1": 1
48             }
49         ],
50         "name": "mobilenet_v2",
```

```
51     "inputSize": 224,  
52     "depthMultiplier": 0.35,  
53     "debug": false,  
54     "hiddenUnits": 0,  
55     "learningRate": 0.005,  
56     "epochs": 100,  
57     "batchSize": 100,  
58     "validationSplit": -0.1,  
59     "batchLogs": "[{\\"batch\\":0,\\"size\\":39,\\"loss  
60     \":3.109633445739746,\\"acc\\":0.07692307978868484},{...}]",  
61     "epochLogs": "[{\\"val_loss\\":2.3543283939361572,\\"val_acc  
62     \":0.25641027092933655,\\"loss\\":3.109633445739746,\\"acc  
63     \":0.07692307978868484},{...}]"  
64 },  
65 "inferenceConfig": {  
66     "roi": {  
67         "x": 0,  
68         "y": 0,  
69         "w": 1,  
70         "h": 1  
71     },  
72     "gpio0func": 0,  
73     "gpio1func": 0  
74 }
```

## Patch Model

Modifizierte einzelne Felder eines Modells.

Endpunkte: `/api/models/{modelID:[0-9a-v]{20}}`, `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}`

Methoden: PATCH

Headers Regexp: `{"Content-Type": "application/json"}`

*Dokumentation folgt*

## Delete Model

Löscht das Modell aus dem Task. Führt zunächst ein Soft-Delete (setzen des Feldes `deletedAt`) aus. Falls bereits Soft-Deleted, wird das Modell hart gelöscht.

Endpunkte: `/api/models/{modelID:[0-9a-v]{20}}`, `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}`

Methoden: DELETE

Rückgaben:

Code	Status	Bedeutung
204	No Content	Erfolgreich gelöscht
400	Bad Request	Task-ID unbekannt oder anderer Fehler

### Infer/Execute Model (single image)

Führt das Model einmalig auf dem aktuellen Bild aus (Inferenz).

Endpunkte: `/api/models/{modelID:[0-9a-v]{20}}/inference`

Methoden: GET

Headers Regexp: `{"Content-Type": "application/json"}`

Exemplarische Rückgabe:

```
1 {
2   "modelID": "c0mh5hv5upvdn1dnhsl0",
3   "type": 1,
4   "roi": {
5     "x": 0,
6     "y": 0,
7     "w": 1,
8     "h": 1
9   },
10  "results": [
11    {
12      "classID": 1,
13      "mappedClassID": 0,
14      "className": "Objekt A",
15      "confidence": 0.3515625,
16      "bbox": {
17        "x": 0.1083346,
18        "y": 0.25184333,
19        "w": 0.73278534,
20        "h": 0.7098198
```

```
21     }
22     },
23     {
24         ...
25     }
26 ]
27 }
```

### Infer/Execute Model (stream)

Startet die Ausführung des Modells auf dem Videostrom. Es werden kontinuierlich Inferenz-Ergebnisse zurückgegeben. Die Rate der Inferenz entspricht dem Minimum aus Bildrate (30 Bilder pro Sekunde) und der Gesamtlatenz aller parallel ausgeführten Modelle. Die Anfrage muss über einen WebSocket gestartet werden (Upgrade).

Endpunkte: `/api/models/{modelID:[0-9a-v]{20}}/inference`, `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}`

Methoden: GET

Exemplarische Rückgabe (WebSocket-Message):

```
1 {
2     "modelID": "c0mh5hv5upvdn1dnhs10",
3     "type": 1,
4     "roi": {
5         "x": 0,
6         "y": 0,
7         "w": 1,
8         "h": 1
9     },
10    "results": [
11        {
12            "classID": 1,
13            "mappedClassID": 0,
14            "className": "Objekt A",
15            "confidence": 0.3515625,
16            "bbox": {
17                "x": 0.1083346,
18                "y": 0.25184333,
19                "w": 0.73278534,
20                "h": 0.7098198
21            }
22        }
23    ]
24 }
```

```
22     },
23     {
24         ...
25     }
26 ]
27 }
```

## Export Model (as zip)

Exportiert das Model als ZIP-Datei.

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}`, `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}/model.zip`

Methoden: `GET`

Headers Regexp: `{("Accept", "application/zip"), {}}`

## Get File of Model

Gibt eine Datei des Models zurück (z.B. `model.json`, `model.tflite`)

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}/{file}`

Methoden: `GET`

## Update File of Model

Endpunkte: `/api/tasks/{taskID:[0-9a-v]{20}}/models/{modelID:[0-9a-v]{20}}/{file}`

Methoden: `PUT`

*Dokumentation folgt*

## Video-spezifische Endpunkte

### H264-Stream (live)

H264-enkodierter WebSocket Stream des Live Kamera-Bildes.

Endpunkte: </api/video/stream/h264>

Methoden: [GET](#)

### **Video Einstellungen**

Endpunkte: </api/video/settings>

Methoden: [GET](#)

### **Video Einstellungen modifizieren**

Endpunkte: </api/video/settings>

Methoden: [PATCH](#)

*Dokumentation folgt*

### **Bild aufnehmen**

Nimmt ein Bild auf und gibt es zurück.

Endpunkte: </api/video/capture>

Methoden: [GET](#)

Rückgabe: PNG

### **Inferenz-spezifische Endpunkte**

Listet die zur Zeit ausgeführten Modelle auf.

Endpunkte: </api/inference/state>

Methoden: [GET](#)

*Dokumentation folgt*

## Einstellungs-spezifische Endpunkte

### GET Netzwerk-Einstellungen

Endpunkte: `/api/settings/network/{iface}`

Methoden: `GET`

*Dokumentation folgt*

### ADD IP-Adresse zur Netzwerk-Schnittstelle

Endpunkte: `/api/settings/network/{iface}/ip`

Methoden: `POST`

*Dokumentation folgt*

### DELETE IP-Adresse von Netzwerk-Schnittstelle

Endpunkte: `/api/settings/network/{iface}/ip`

Methoden: `DELETE`

*Dokumentation folgt*

### SW-Update der Kamera

Endpunkte: `/api/settings/update`

Methoden: `POST`

*Dokumentation folgt*

### GET Autostart Modelle

Gibt die Modelle zurück, welche beim Start (Boot) der Kamera automatisch gestartet werden.

Endpunkte: `/api/settings/models/autostart`

Methoden: `GET`

Headers Regexp: `{"Content-Type": "application/json"}`

*Dokumentation folgt*



### **Füge Autostart Modell hinzu**

Endpunkte: `/api/settings/models/autostart`

Methoden: `POST`

*Dokumentation folgt*

### **DELETE Autostart Model**

Endpunkte: `/api/settings/models/autostart/{modelID:[0-9a-v]{20}}`

Methoden: `DELETE`

*Dokumentation folgt*

### **GET Hardware Status**

Beispielsweise Temperaturen der CPU.

Endpunkte: `/api/settings/hardware/status`

Methoden: `GET`

Headers Regexp: `{"Content-Type": "application/json"}`

*Dokumentation folgt*

## **Status-spezifische Endpunkte**

### **GET Service Status**

Endpunkte: `/api/service/{service}`

Methoden: `GET, POST`

*Dokumentation folgt*

### **RPC Service**

Endpunkte: `/api/service/{service}/{command}`

Methoden: `GET`

*Dokumentation folgt*