

Task REST API V1

ValueStreamer GmbH

Content

1. Introduction

- 1.1 REST-API & basic technical knowledge
- 1.2 Authorization and user assignment for changes via the API
- 1.3 Task board overview

2. Necessary metadata for querying, creating and updating tasks

- 2.1 Metadata: Use cases and API endpoints
- 2.2 Meta data REST endpoint details and examples

3. Task board use cases and REST-API endpoints

- 3.1 Use case and REST-API endpoint mapping
- 3.2 Fields of a task and where to get them
- 3.3 Fields of a task and where to insert or update them
- 3.4 REST endpoints details and examples

1. Introduction

1.1 REST-API & basic technical knowledge

- The Task API allows clients to fetch or insert task entries via **REST** and **JSON**
- The API specification contains all endpoints and JSON formats in **.yaml** files as **OpenAPI 3** format
- We recommend to view the file via **Swagger UI** / Editor.
- All access to the API is done via **HTTPS**
- The API uses appropriate **HTTP methods** for each action.
 - GET: For retrieving one or more tasks, meta data or a task's description
 - POST: For creating tasks
 - PUT: For updating a task or a task's description
 - DELETE: For deleting a task
- The API returns corresponding **HTTP status codes** for each request, see Swagger file. E.g.
 - 2xx for success: 200 OK, 204 Deleted
 - 4xx for client errors: 400 Bad request, 401 Unauthorized, 404 Not found
 - 5xx for server errors: 500 Internal server error
- The API is reachable via <https://api-<tenant>.valuestreamer.de/api/> where <tenant> must be replaced with your tenant
- E.g.: <https://api-demo.valuestreamer.de/api/exchange/tasks>

Task			^
GET	/exchange/tasks	Retrieve a collection of tasks.	▼ 🔒
POST	/exchange/tasks	Create a task	▼ 🔒
GET	/exchange/tasks/{id}	Retrieve a specific task.	▼ 🔒
PUT	/exchange/tasks/{id}	Update a task	▼ 🔒
DELETE	/exchange/tasks/{id}	Delete a specific task.	▼ 🔒
Task: Description			^
Endpoints for fetching and updating the data of the richtext input field 'description'.			
GET	/exchange/tasks/{id}/description	Get the description.	▼ 🔒
PUT	/exchange/tasks/{id}/description	Update the description.	▼ 🔒
Task: Status history			^
Endpoints for fetching the status history of a task.			
GET	/exchange/tasks/{id}/status-history	Get the status history of the task, ordered by date (desc.)	▼ 🔒
Meta			^
Endpoints to retrieve meta data for tasks.			
GET	/exchange/taskboard-categories	Get all available categories on the tenant	▼ 🔒
GET	/exchange/taskboard-labels	Get all available labels on the tenant	▼ 🔒
GET	/exchange/teams/{id}/taskboard	Get the details of a taskboard for a specific team: Available labels and categories	▼ 🔒

1.2 Authorization and user assignment

Authorization

1. Authorization on the API endpoints works via basic auth per request
2. ValueStreamer Support provides username + password to the customer
3. When sending requests to the API, the basic auth credentials must be passed base64 encoded in the authorization header like this:
Authorization: Basic 123456

Assigned user for task changes via the API

1. The ValueStreamer Support creates a ValueStreamer Application user named "API-USER"
2. Whenever a task is created, modified or deleted via the API, this API-USER is written to the changelogs (createdBy, modifiedBy, ...)

HTTP Example:

```
GET /api/exchange/tasks?include=ALL&team=c2c1483f-7b29-4ed5-9d46-c290f8cb50a5
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic
a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTdLdXY=
```

Postman example:

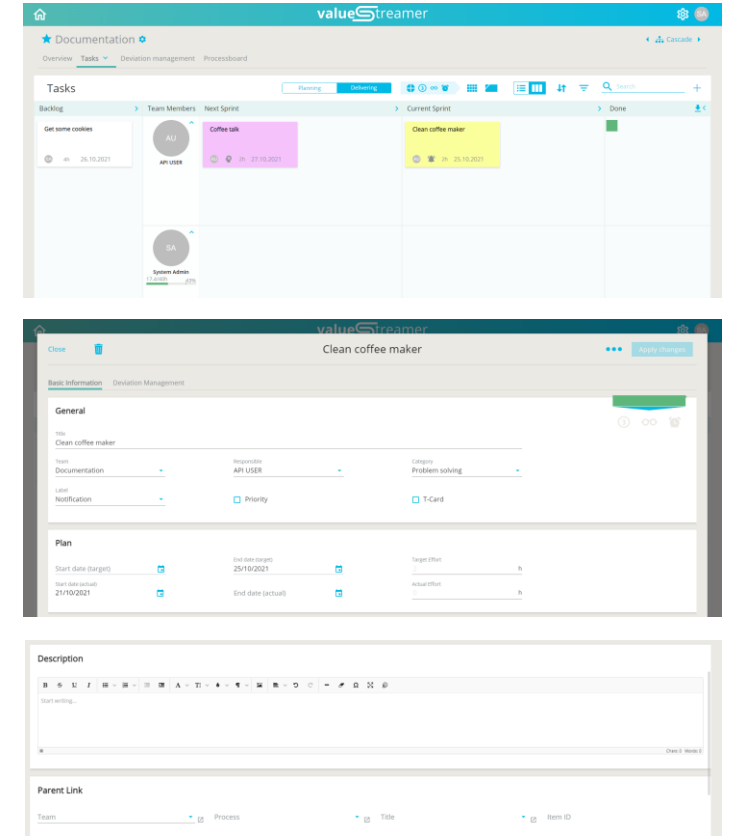
The screenshot shows the Postman interface for a GET request. The URL is `{{url}}/api/exchange/tasks?include=ALL&team=c2c1483f-7b29-4ed5-9d46-c290f8cb50a5...`. The 'Auth' tab is selected, showing 'Basic Auth' as the type. The 'Username' field contains `{{basicUser}}` and the 'Password' field contains masked characters. A 'Show Password' checkbox is present and unchecked. Below the fields, a note states: 'The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)'. The top bar shows 'GET' and a 'Send' button. The bottom bar shows 'Params', 'Auth', 'Headers (8)', 'Body', 'Pre-req.', 'Tests', and 'Settings'.

Additional information

Created by API USER	Created on 19.10.2021	Finished on -
Changed by System Admin	Changed on 20.10.2021	Archived on -

1.3 Task board overview (1)

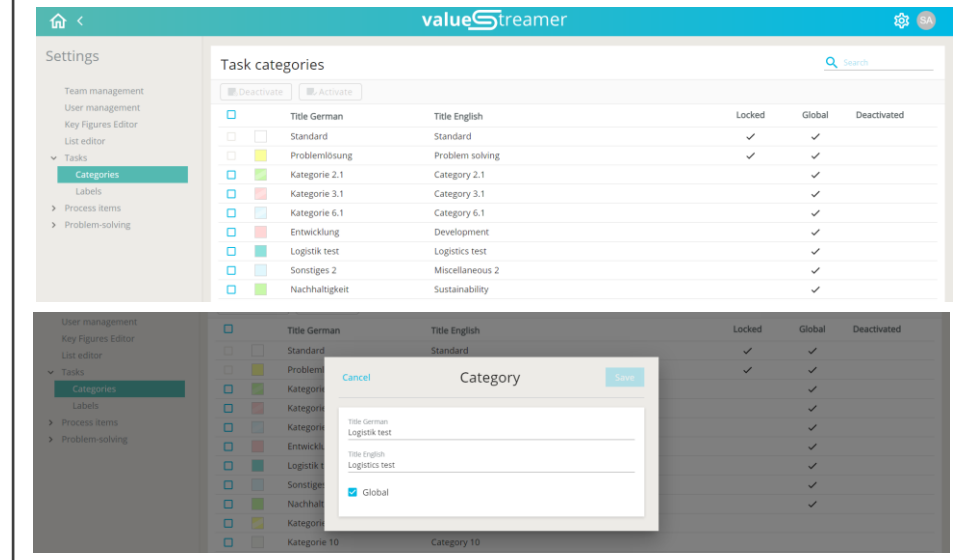
- The task board view is available for every team by default
- Each team member can create as many tasks as they want in the backlog. They also can change the progress by pulling their tasks in the corresponding column
- The progress of a task can take the following value:
 - CREATED: Task in backlog
 - PLANNED: Task is in the next sprint
 - INPROGRESS: Task is in the current sprint
 - DONE: Task is done
 - ARCHIVED: Task is archived, read only and only visible in the task list -> Archived
- After a task has the progress “DONE” it can be archived via user interface. This impacts the KPI hit rate and level of disturbance and is not possible to do via API
- A task can have an assignee as responsiblePerson, who must be a member of the team
- The fields “Category”, “Priority”, “Title”, “Target End Date” and “Target Effort” are **mandatory**
- Based on the list of available categories of the team, a category must be assigned to a task
- Based on the list of available labels of the team, a label can be assigned to a task
- A task has a status. The status can be green, blue or red. If the status color is changed, a reason must be provided as status message
- A task can be cascaded which means that the task is created from one team and assigned to another team as responsibleTeam
- Tasks can be of type T-Card. It is possible to create tasks of type T-Card via API, although not all features of T-Cards are available using the API
- When moving T-card tasks to DONE, they can be re-planned via user interface. This is not possible via API
- A task can be linked to a process activity



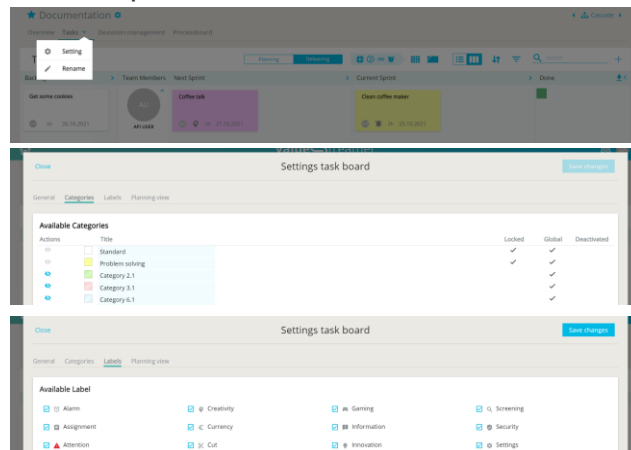
1.3 Task board overview (2)

- The ValueStreamer application provides a set of task categories and labels
- The system admin has the following options to customize the categories and labels:
 - Activate or deactivate entries
 - Rename entries
 - Set categories to global. Global categories can't be renamed via team-specific task board administration
- The team admin can choose which categories and labels are available in the teams' task board via the task board administration. Locked categories can't be deselected or renamed
- The team admin can overwrite a category name for his team, if the category is not locked
- When creating a task, a task is always assigned a category, optionally also a label

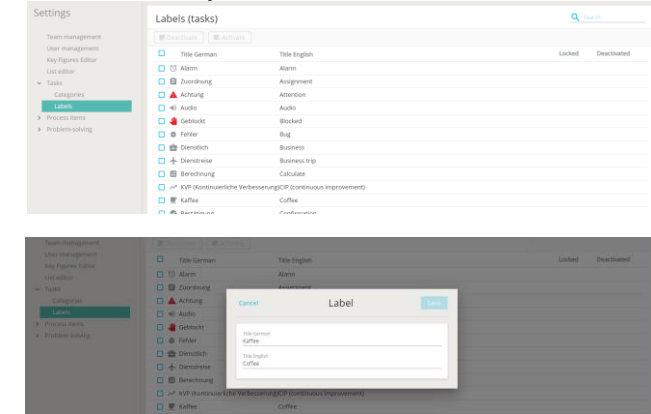
Screens: Task category system administration



Screens: Team-specific task board administration



Screens: Task label system administration



2. Necessary meta data for querying, creating and updating tasks

2.1 Meta data: Use cases and API endpoints

- The following table describes how to get all the data you need for interactions with the API

#	Use case	Use case description	HTTP method	API URL
1	Team: Get the list of teams	Returns all teams with id and name. The id can be used to fetch the team details.	GET	/exchange/teams
2	Team: Get a team's details (by team id)	Returns the details of a team by its team id: <ul style="list-style-type: none">▪ Team's name and short name▪ List of team members (potential assignees resp. responsible users)	GET	/exchange/teams/{id}
3	Get the taskboard details of a team	Returns the task board details of the given team with available labels and categories	GET	/exchange/teams/{id}/taskboard
4	Get the list of all users	Returns all existing users with id and name.	GET	/exchange/users
5	Get the list of all categories	Returns all available categories on the tenant	GET	/exchange/taskboard-categories
6	Get the list of all labels	Returns all available labels on the tenant	GET	/exchange/taskboard-labels

For more details, please have a look into the OpenAPI files.

2.2 #1 GET /exchange/teams

Get the list of teams

- The endpoint returns a lists of all teams present on the tenant
- Media type: application/json

Request:
GET /api/exchange/teams HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdldXY=

Response:

```
[
  {
    "id": "5bcaa97a-d0c1-11ea-87d0-0242ac130003",
    "name": "Linie 1",
    "shortName": "Linie 1",
    "parentTeam": "5bcaa8b2-d0c1-11ea-87d0-0242ac130003"
  },
  {
    "id": "5bcaaa38-d0c1-11ea-87d0-0242ac130003",
    "name": "Linie 2",
    "shortName": "Linie 2",
    "parentTeam": "5bcaa8b2-d0c1-11ea-87d0-0242ac130003"
  },
  {
    "id": "d72fb249-af46-43d8-a116-afd5b32815ef",
    "name": "Log-X",
    "shortName": "Log-X",
    "parentTeam": "14321ce0-ba75-4336-ab26-1b13ebc33114"
  },
  {
    "id": "5bca9930-d0c1-11ea-87d0-0242ac130003",
    "name": "Management",
    "shortName": "Management",
    "parentTeam": "8ba7de30-41b8-4fb3-8d19-dd6c9e92ead7"
  }
]
```

2.2 #2 GET /exchange/teams/{id}

Get team details

- The endpoint returns the details of a team, like the team's members
- Media type: application/json

Request:
GET /api/exchange/teams HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9wTDdldXY=

Response:

```
{
  "id": "5bcaa97a-d0c1-11ea-87d0-0242ac130003",
  "name": "Linie 1",
  "shortName": "Linie 1",
  "parentTeam": "5bcaa8b2-d0c1-11ea-87d0-0242ac130003",
  "members": [
    {
      "id": "e3373620-d0c2-11ea-87d0-0242ac130003",
      "firstName": "Nino",
      "lastName": "Baresi",
      "email": "Nino.Baresi@valuestreamer-demo.de"
    },
    {
      "id": "e3375560-d0c2-11ea-87d0-0242ac130003",
      "firstName": "Jürgen",
      "lastName": "Seiler",
      "email": "Jürgen.Seiler@valuestreamer-demo.de"
    }
  ],
  "processBoards": [
    {
      "id": "78f748e5-0ad5-4f1a-951e-7bcc7cc94d97",
      "name": "log-x"
    },
    {
      "id": "d40f30ae-f509-4126-bd71-73da5df21967",
      "name": "KVP Projekte"
    }
  ]
}
```

2.2 #3 GET /exchange/teams/{id}/taskboard

Get the task board details of a team

- The endpoint returns all available labels and categories of the given team
- The sections of labels and categories have the same structure as given by the endpoints /exchange/taskboard-labels and /exchange/taskboard-categories
- **Note:** Since version 1.0.2 of the API, the property „tCard“ is always false
- Media type: application/json

```
Request:
GET /api/exchange/teams/{id}/taskboard HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTdDldXY=
```

```
Response:
{
  'categories': [
    {
      'id': 'ad1f7f3b-3a00-3400-3400-000064006500',
      'name': [
        { 'language': 'EN', 'name': 'Dark blue' },
        { 'language': 'DE', 'name': 'Dunkel Blau' }
      ],
      'color': '#D5DAF8',
      'colorStripes': null,
      'locked': false,
      'default': false,
      'global': false,
      'problemSolving': false
    }, ...
  ],
  'labels': [
    {
      'id': '87e28661-847d-4f61-afe7-6933732cfc5d',
      'name': [
        { 'language': 'EN', 'name': 'Strategic task' },
        { 'language': 'DE', 'name': 'Strategische Aufgabe' }
      ],
      'removable': true,
      'tCard': false
    }, ...
  ]
}
```

2.2 #4 GET /exchange/users

Get the list of users

- The endpoint returns all users present on the tenant
- Media type: application/json

Request:
GET /api/exchange/users HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTdDldXY=

Response:

```
[
  {
    "id": "e3372d60-d0c2-11ea-87d0-0242ac130003",
    "firstName": "Daniel",
    "lastName": "Ackermann",
    "email": "Daniel.Ackermann@valuestreamer-demo.de"
  },
  {
    "id": "e3372fcc-d0c2-11ea-87d0-0242ac130003",
    "firstName": "Martina",
    "lastName": "Amsel",
    "email": "Martina.Amsel@valuestreamer-demo.de"
  },
  {
    "id": "e3373314-d0c2-11ea-87d0-0242ac130003",
    "firstName": "Sarah",
    "lastName": "Andersen",
    "email": "Sarah.Andersen@valuestreamer-demo.de"
  },
  ...
]
```

2.2 #5 GET /exchange/taskboard-categories

Get all available categories

- The endpoint returns a list of all available task categories on the tenant
- Every category has an ID, names in German and English, and entry specific parameters for
 - default Marker for the default category, which is available in every task board
 - global Globally available categories are present in every task board
 - locked Locked categories can't be renamed
 - problemSolving Tasks that are created for a countermeasure are automatically assigned the category problemSolving. Such tasks can be updated via the API, but no new tasks can be created with this category via the API
- The response also contains design parameters
 - color Background color
 - colorStripes Background stripes color
- Media type: application/json

```
Request:
GET /api/exchange/taskboard-categories HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdLdXY=
```

```
Response:
[
  {
    'id': '8071a314-473f-467b-be9c-f9ac3405000f',
    'name': [
      {
        'language': 'EN',
        'name': 'Category 9'
      },
      {
        'language': 'DE',
        'name': 'Kategorie 9'
      }
    ],
    'color': '#FFD5B8',
    'colorStripes': null,
    'locked': false,
    'default': false,
    'global': false,
    'problemSolving': false
  }, ...
]
```

2.2 #6 GET /exchange/taskboard-labels

Get all the labels available

- The endpoint returns all available task labels on the tenant
- Every label has an ID and names in German and English
- The property tCard shows whether the task is of type tCard or not.
Note: Since version 1.0.2 of the API, this property is always false
- The property removable defines, whether the label can be deselected by the team admin from his taskboard (removable true) or not (removable false)
- Media type: application/json

```
Request:
GET /api/exchange/taskboard-labels HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdLdXY=
```

```
Response:
[
  {
    'id': '87e28661-847d-4f61-afe7-6933732cfc5d',
    'name': [
      {'language': 'EN', 'name': 'Strategic task'},
      {'language': 'DE', 'name': 'Strategische Aufgabe'}
    ],
    'removable': true,
    'tCard': false
  },...
]
```

3. Task board use cases and REST-API endpoints

3.1 Use case and REST-API endpoint mapping

#	Use case	HTTP Method	API URL
1	Retrieve a collection of tasks according to the query parameters	GET	/exchange/tasks?{query-parameters}
2	Retrieve the given task and each of its fields	GET	/exchange/tasks/{task-id}
3	Create a new task	POST	/exchange/tasks
4	Update the given task	PUT	/exchange/tasks/{task-id}
5	Delete the given task	DELETE	/exchange/tasks/{task-id}
6	Get the description of the given task	GET	/exchange/tasks/{task-id}/description
7	Update the description of the given task	PUT	/exchange/tasks/{task-id}/description
8	Get the status history of the given task	GET	/exchange/tasks/{task-id}/status-history

For more details, please have a look into the OpenAPI Files.

3.2 Fields of a task and where to get them as response

Screen 1: Task modal view

Close

Get some cookies

...

Apply changes

Basic Information

Deviation Management

General

Title

Get some cookies

Team

Documentation

Responsible

System Admin

Category

Standard

Label

Priority

T-Card

Plan

Start date (target)

26/10/2021

End date (target)

26/10/2021

Target Effort

4

h

Start date (actual)

End date (actual)

Actual Effort

0

h

Screen2 : Task modal view – Status dropdown

Basic Information

Deviation Management

General

Title

Get some cookies

Team

Documentation

Responsible

System Admin

Category

Standard

Label

Priority

T-Card

Status

Cancel

Save

Frank will bring some

Status history

List of endpoints:							
				1	2	3	4
#	Field	Format	JSON field name				
	Task id	UUID	id	✓	✓		
1	Title	String	title	✓	✓		
2	Status	Int{1,2,3}	status	✓	✓		✓
3	Reason for status change	String	statusMessage		✓		✓
4	Team	UUID	responsibleTeam	✓	✓		
5	Responsible (user)	UUID	responsiblePerson	✓	✓		
6	Category	UUID	category	✓	✓		
7	Label	UUID	label	✓	✓		
8	Priority	Boolean	priority	✓	✓		
9	T-Card	Boolean	tCard		✓		
10	Start date (target)	String	targetStartDate		✓		
11	End Date (target)	String	targetEndDate	✓	✓		
12	Target Effort	Float	targetEffort	✓	✓		
13	Start date (actual)	String	actualStartDate		✓		

3.2 Fields of a task and where to get them as response

Screen 3: Task modal view

Close [icon] Get some cookies [icon] [icon] [icon] Apply changes

Basic Information Deviation Management

General

Title
Get some cookies

Team
Documentation

Responsible
System Admin

Category
Standard

Label

Priority

T-Card

Plan

Start date (target) 26/10/2021

End date (actual) 14

Target Effort

Actual Effort 15

Description

Start writing... 16

Parent Link

Team Demo 17

Process Processboard 18

Title Process test 19

Item ID 19

Additional information

Created by System Admin 20

Created on 21.10.2021 21

Finished on

Changed by System Admin 23

Changed on 25.10.2021 24

Archived on

25

Screen 4: Task board progress columns

Tasks 27

27 Planning Delivering 27

Backlog Team Members Next Sprint Current Sprint Done

#	Field	Format	JSON field name	List of endpoints:			
				1. GET /exchange/tasks/{query_parameters}	2. GET /exchange/tasks/{task-id}	3. GET /exchange/tasks/{task-id}/description	4. GET /exchange/tasks/{task-id}/status-history
				1	2	3	4
14	End date (actual)	String	actualEndDate		✓		
15	Actual Effort	Float	actualEffort		✓		
16	Description	HTML String				✓	
17	Process - Team	UUID	linkedToTeam		✓		
18	Process - Board	UUID	linkedToProcess		✓		
19	Process - Title/Item ID	UUID	linkedToProcessItem		✓		
20	Created by (user)	String	createdBy		✓		
21	Created on	String	createdDate		✓		
22	Finished on	String	actualEndDate		✓		
23	Changed by (user)	String	modifiedBy		✓		✓
24	Changed on	String	modifiedDate		✓		✓
25	Archived on	String	archivedDate		✓		
26	Cascading team (not visible)	UUID	createdByTeam	✓	✓		
27	Progress	String enum	progress	✓	✓		

3.3 Fields of a task and where to insert or update them

Screen 1: Task modal view

Close

Get some cookies

Apply changes

Basic Information

Deviation Management

General

Title

Get some cookies

Team

Documentation

Responsible

System Admin

Category

Standard

Label

Priority

T-Card

Plan

Start date (target)

End date (target)

26/10/2021

Target Effort

h

Start date (actual)

End date (actual)

Actual Effort

h

Screen2 : Task modal view – Status dropdown

Basic Information

Deviation Management

General

Title

Get some cookies

Team

Documentation

Responsible

System Admin

Category

Standard

Label

Priority

T-Card

Status

Cancel

Save

Frank will bring some

Status history

#	Field	Format	JSON field name	List of endpoints:		
				1. POST /exchange/tasks	2. PUT /exchange/tasks/{task-id}	3. PUT /exchange/tasks/{task-id}/description
				1	2	3
	Task id	UUID	id			
1	Title	String	title	✓	✓	
2	Status	Int{1,2,3}	status	✓	✓	
3	Reason for status change	String	statusMessage	✓	✓	
4	Team	UUID	responsibleTeam	✓	✓	
5	Responsible (user)	UUID	responsiblePerson	✓	✓	
6	Category	UUID	category	✓	✓	
7	Label	UUID	label	✓	✓	
8	Priority	Boolean	priority	✓	✓	
9	T-Card	Boolean	tCard	✓	✓	
10	Start date (target)	String	targetStartDate	✓	✓	
11	End Date (target)	String	targetEndDate	✓	✓	
12	Target Effort	Float	targetEffort	✓	✓	
13	Start date (actual)	String	actualStartDate	Is automatically set when the task has the progress INPROGRESS		

1/28/2026

valueStream

20

3.3 Fields of a task and where to insert or update them

Screen 3: Task modal view

[Close](#) Get some cookies [Apply changes](#)

Basic Information Deviation Management

General

Title
Get some cookies

Team Documentation Responsible System Admin Category Standard

Label ☐ Priority ☐ T-Card

Plan

Start date (target) 26/10/2021 Target Effort 4 h

Start date (actual) End date (actual) 14 Actual Effort 15 h

Description

Start writing... 16

Chars: 0 Words: 0

Parent Link

Team Demo Process Processboard Title Process test Item ID 19

Additional information

Created by System Admin Created on 21.10.2021 Finished on -

Changed by System Admin Changed on 25.10.2021 Archived on -

Screen 4: Task board progress columns

#	Field	Format	JSON field name	List of endpoints: 1. POST /exchange/tasks 2. PUT /exchange/tasks/{task-id} 3. PUT /exchange/tasks/{task-id}/description		
				1	2	3
14	End date (actual)	String	actualEndDate	Is automatically set when the task has the progress DONE		
15	Actual Effort	Float	actualEffort	✓	✓	
16	Description	HTML String				✓
17	Process - Team	UUID	linkedToTeam			
18	Process – Board	UUID	linkedToProcess			
19	Process - Title/Item ID	UUID	linkedToProcessItem			
20	Created by (user)	String	createdBy			
21	Created on	String	createdDate			
22	Finished on	String	actualEndDate	Is automatically set when the task has the progress DONE		
23	Changed by (user)	String	modifiedBy			
24	Changed on	String	modifiedDate			
25	Archived on	String	archivedDate			
26	Cascading team (not visible)	UUID	createdByTeam	✓		
27	Progress	String enum	progress	✓	✓	

3.4 #1 GET /exchange/tasks?{query-parameters}

Retrieve a collection of tasks according to the query parameters

- The endpoint returns all tasks that match the given query parameters
- Filter attributes as req. parameters:
 - **&team=** → id of the team
 - **&include=** → ALL, CASCADED or RESPONSIBLE
 - RESPONSIBLE: Only include tasks that the team is responsible for
 - CASCADED: Only include tasks, that were cascaded by the team. Tasks are cascaded when they were created in the requested team (property createdByTeam), but have a different responsibleTeam
 - ALL: Include both, RESPONSIBLE + CASCADED tasks
 - **&progress=** → Filter by task progress
 - Available values : CREATED, PLANNED, INPROGRESS, DONE, ARCHIVED
 - A single value or a comma separated list can be provided, e.g. CREATED,INPROGRESS
 - **&page=1** → Page number, max. 200 task entries per page
- Accept: application/json

Bold = required / mandatory

```
Request:
GET /api/exchange/tasks?team=49af7f6c-c1e5-49c7-99ed-
c27b29067eec&include=ALL&progress=CREATED,INPROGRESS&include=ALL&page=1
HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdLdXY=
```

```
Response:
{
  'page':
    {
      'page': 1,
      'elementsOnPage': 1,
      'elementsPerPage': 200,
      'totalPages': 1,
      'totalElements': 1
    },
  'entries': [
    {
      'id': '81beaa9b-b510-4924-af84-0fe651de8fcb',
      'category': '263c2d97-762f-4ef3-a350-b88225d00885',
      'progress': 'INPROGRESS',
      'priority': false,
      'label': null,
      'responsiblePerson': '190cd123-08c2-496d-8f39-6dc31dd706c8',
      'responsibleTeam': '49af7f6c-c1e5-49c7-99ed-c27b29067eec',
      'createdByTeam': '49af7f6c-c1e5-49c7-99ed-c27b29067eec',
      'status': 2,
      'title': 'Clean coffee maker',
      'targetEndDate': '2021-10-29',
      'targetEffort': 2.0
    },...
  ]
}
```

3.4 #2 GET /exchange/tasks/{task-id}

Retrieve the given task and each of its fields

- The endpoint returns all fields and additional information of the given task
- The description of the task can be retrieved with endpoint #6
- Accept: application/json
- The tCard field indicates whether this is a T-Card task. When a T-Card is created or updated via the Exchange API with tCard set to true, the system automatically manages a recurrence pattern (set to WEEKLY) internally, though this recurrence pattern field is not exposed in the API response.

Request:

```
GET /api/exchange/tasks/8377539f-52e5-4637-aac5-6d0a7665379d HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdldXY=
```

Response:

```
{
  "id": "d55ed0f1-b77d-43f2-8011-8e4f8522feed",
  "actualEffort": null,
  "actualEndDate": null,
  "actualStartDate": null,
  "archivedDate": null,
  "category": "263c2d97-762f-4ef3-a350-b88225d00881",
  "createdBy": "190cd123-08c2-496d-8f39-6dc31dd706c8",
  "createdByTeam": "49af7f6c-c1e5-49c7-99ed-c27b29067eec",
  "createdDate": "2021-11-17",
  "label": "7c0d82b5-b943-4576-8a86-438de327ab10",
  "linkedToProcess": null,
  "linkedToProcessitem": null,
  "linkedToTeam": null,
  "modifiedBy": null,
  "modifiedDate": null,
  "priority": true,
  "progress": "CREATED",
  "responsiblePerson": null,
  "responsibleTeam": "49af7f6c-c1e5-49c7-99ed-c27b29067eec",
  "status": 2,
  "statusMessage": "",
  "targetEffort": 6.6,
  "targetEndDate": "2021-11-01",
  "targetStartDate": null,
  "title": "Buy coffee beans",
  "tCard": true
}
```

3.4 #3 POST /exchange/tasks

Create a new task

- The endpoint creates a new task
- The task ID will be generated
- Property: category
 - Based on the list of available categories of the team, a category ID must be provided
 - It is not possible to use the category “problem solving” as category when creating tasks, as this category is reserved for tasks which are derived from countermeasures
- Property: createdByTeam
 - This property can be used to define that this task was cascaded from the specified team to the responsible team
- Property: label
 - Based on the list of available labels of the team, a label ID can be added. The label restriction has been removed - any label can now be used for T-Cards.
- Property: progress
 - Progress is set to CREATED when no value is provided
 - Progress can be set to CREATED, PLANNED, INPROGRESS or DONE
 - It is not possible to set the progress ARCHIVED, as tasks can be archived only via task board user interface
- Property: status
 - Value must be either 1 (green), 2 (blue) or 3 (red)
- Property: T-Card
 - If a recurring T-card task should be created, the property tCard must be set to TRUE. When tCard is set to TRUE, the system automatically sets the recurrence pattern to WEEKLY. When tCard is set to FALSE: recurrence pattern is set to null.
- Accept: application/json – Content-Type: application/json

Bold = required / mandatory

Request:

POST /api/exchange/tasks HTTP/1.1

Host: api-demo.valuestreamer.de

Accept: application/json

Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdldXY=

```
{
  'category': '263c2d97-762f-4ef3-a350-b88225d00882',
  'priority': true,
  'responsibleTeam': '49af7f6c-c1e5-49c7-99ed-c27b29067eec',
  'status': 2,
  'targetEndDate': '2021-11-01',
  'targetEffort': 6.6,
  'title': 'New task'
}
```

Response:

```
{
  'id': '248f4a43-ccd0-448a-89d1-25e82ce15502',
  'targetEndDate': '2021-11-01',
  'targetStartDate': null,
  'targetEffort': 6.6,
  'actualEndDate': null,
  'actualStartDate': null,
  'actualEffort': 3.0,
  'archivedDate': null,
  'category': '263c2d97-762f-4ef3-a350-b88225d00882',
  'createdBy': '190cd123-08c2-496d-8f39-6dc31dd706c8',
  'createdByTeam': '49af7f6c-c1e5-49c7-99ed-c27b29067eec',
  'createdDate': '2021-10-25',
  'linkedToTeam': null,
  'linkedToProcessitem': null,
  'linkedToProcess': null,
  'modifiedBy': null,
  'modifiedDate': null,
  'priority': true,
  'label': null,
  'responsiblePerson': null,
  'responsibleTeam': '49af7f6c-c1e5-49c7-99ed-c27b29067eec',
  'progress': 'CREATED',
  'status': 2,
  'statusMessage': '',
  'title': 'New task',
  'tCard': false
}
```


3.4 #4 PUT /exchange/tasks/{task-id}

Update the given task

- The endpoint updates the given task
- targetEndDate, targetEffort, category, priority, responsibleTeam, status and title are **mandatory**
- If properties are sent with NULL values, the existing values in the database are removed resp. overwritten with NULL
- If the status is updated to another value, e.g. from 1 (green) to 3 (red), a new status history entry is created. When the status is changed, the field „statusMessage“ has to be provided and must be not null or empty
- Accept: application/json – Content-Type: application/json
- When the tCard property is updated, the system automatically manages the recurrence pattern:
 - If tCard is changed from false to true: Recurrence pattern is automatically set to WEEKLY
 - If tCard is changed from true to false: Recurrence pattern is set to null
- Since version 1.0.2 of the API, the label restriction has been removed - any label can now be used for T-Cards

```
Request:
PUT /api/exchange/tasks/ab913e1e-7581-4269-8513-fb8067b4d613 HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTdDldXY=
{
  "category": "263c2d97-762f-4ef3-a350-b88225d00881",
  "label": null,
  "priority": false,
  "progress": "CREATED",
  "responsiblePerson": "b3b2b4c6-fa90-4f1b-95e5-a06e80859464",
  "responsibleTeam": "c2c1483f-7b29-4ed5-9d46-c290f8cb50a5",
  "status": 1,
  "statusMessage": "",
  "targetEffort": 2.0,
  "targetEndDate": "2021-11-18",
  "targetStartDate": null,
  "title": "Coffee talk",
  "tCard": false
}
```

```
Response:
{
  "id": "ab913e1e-7581-4269-8513-fb8067b4d613",
  "actualEffort": 0.0,
  "actualEndDate": null,
  "actualStartDate": null,
  "archivedDate": null,
  "category": "263c2d97-762f-4ef3-a350-b88225d00881",
  "createdBy": "a2950808-8654-4586-b37b-4af997efc096",
  "createdByTeam": "c2c1483f-7b29-4ed5-9d46-c290f8cb50a5",
  "createdDate": "2021-11-17",
  "label": null,
  "linkedToProcess": null,
  "linkedToProcessitem": null,
  "linkedToTeam": null,
  "modifiedBy": "190cd123-08c2-496d-8f39-6dc31dd706c8",
  "modifiedDate": "2021-11-17",
  "priority": false,
  "progress": "CREATED",
  "responsiblePerson": "b3b2b4c6-fa90-4f1b-95e5-a06e80859464",
  "responsibleTeam": "c2c1483f-7b29-4ed5-9d46-c290f8cb50a5",
  "status": 1,
  "statusMessage": "",
  "targetEffort": 2.0,
  "targetEndDate": "2021-11-18",
  "targetStartDate": null,
  "title": "Coffee talk",
  "tCard": false
}
```

3.4 #5 DELETE /exchange/tasks/{task-id}

Delete a task

- The endpoint deletes the given task
- When a task is deleted, entries of the following types related to the task are also deleted:
 - Deviation
 - Countermeasure
 - Go + see
 - Top issue
 - Feedback
- Accept: application/json

Request:
DELETE /exchange/tasks/54e0f2ab-9c22-402f-a915-f63b4c4b7071 HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdLdXY=

Response:

3.4 #6 GET /exchange/tasks/{task-id}/description /exchange/tasks/{task-id}/description

- The endpoint returns the description of the given task as HTML code
- **Accept:** text/html

Request:
GET /api/exchange/tasks/81beaa9b-b510-4924-af84-0fe651de8fcb/description
HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: text/html
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdldXY=

Response:
<p>Your text could stand here</p>

3.4 #7 PUT /exchange/tasks/{task-id}/description

Update the description of the given task

- The endpoint updates the description of a task
- Valid HTML code must be sent in the request body
- **Accept: text/html** – **Content-Type: text/html**

Request:
PUT /api/exchange/tasks/54e0f2ab-9c22-402f-a915-f63b4c4b7071/description
HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: 'text/html'
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTdDldXY=

'<p>Change</p>'

Response:
'<p>Change</p>'

3.4 #8 GET /exchange/tasks/{task-id}/status-history

Get the status history of the given task

- The endpoint returns the status history of the given task
- The status history contains the list of status updates ordered by date
- An entry contains the following properties:
 - comment: contains the statusMessage resp. reason for the status update
 - modifiedDate: when the status was updated
 - modifiedByUser: id of the person who updated the status
 - status: the new status value
- Accept: application/json

```
Request:
GET /api/exchange/tasks/81beaa9b-b510-4924-af84-0fe651de8fcb/status-history
HTTP/1.1
Host: api-demo.valuestreamer.de
Accept: application/json
Authorization: Basic a3BpZXhjaGFuZ2U6NW1VZE1HbWpxOTNvdE9WTDdLdXY=
```

```
Response:
[
  {
    'comment': 'Status updated',
    'modifiedByUser': 'a2950808-8654-4586-b37b-4af997efc096',
    'modifiedDate': '2021-10-25',
    'status': 1
  },
  {
    'comment': 'Status updated',
    'modifiedByUser': 'a2950808-8654-4586-b37b-4af997efc096',
    'modifiedDate': '2021-10-24',
    'status': 2
  }
]
```

valueSstreamer

ECHTE ZUSAMMENARBEIT