

# Platform Base APIs

---

# ClusterModule [cluster.alauda.io/v1alpha1]

## Description

ClusterModule is the Schema for the clustermodules API

## Type

object

## Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources</a>

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds</a>
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	ClusterModuleSpec defines the desired state of ClusterModule
<code>status</code>	<code>object</code>	ClusterModuleStatus defines the observed state of ClusterModule

## .spec

### Description

ClusterModuleSpec defines the desired state of ClusterModule

### Type

`object`

Property	Type	Description
<code>parallelUpgrade</code>	<code>boolean</code>	ParallelUpgrade stands for if the upgrading could execute parallely.
<code>upgradeModulesInfos</code>	<code>array</code>	UpgradeModuleInfos stands for the information to upgrade the modules
<code>valuesOverride</code>	<code>object</code>	ValuesOverride ...
<code>version</code>	<code>string</code>	

## `.spec.upgradeModulesInfos`

### Description

UpgradeModuleInfos stands for the information to upgrade the modules

### Type

`array`

## `.spec.upgradeModulesInfos[]`

### Description

UpgradeModuleInfo ...

### Type

`object`

### Required

`name` `version`

Property	Type	Description
config	object	
name	string	
version	string	

## .spec.upgradeModulesInfos[].config

### Type

object

## .spec.valuesOverride

### Description

ValuesOverride ...

### Type

object

## .status

### Description

ClusterModuleStatus defines the observed state of ClusterModule

### Type

object

### Required

editable upgradable

Property	Type	Description
appReleases	object	

Property	Type	Description
<code>base</code>	<code>object</code>	ModuleStatus ...
<code>editable</code>	<code>boolean</code>	
<code>needUpgrade</code>	<code>boolean</code>	
<code>sync</code>	<code>string</code>	SyncMethod ...
<code>unableUpgradeReason</code>	<code>string</code>	
<code>upgradable</code>	<code>boolean</code>	

## `.status.appReleases`

### Type

`object`

## `.status.base`

### Description

ModuleStatus ...

### Type

`object`

### Required

`deployStatus` `runningStatus`

Property	Type	Description
<code>appReleases</code>	<code>array</code>	

Property	Type	Description
<code>deployStatus</code>	<code>string</code>	DeployStatus ...
<code>deployTime</code>	<code>string</code>	
<code>lastProbeTime</code>	<code>string</code>	
<code>lastTransitionTime</code>	<code>string</code>	
<code>logPath</code>	<code>string</code>	
<code>message</code>	<code>string</code>	
<code>reason</code>	<code>string</code>	
<code>runningStatus</code>	<code>string</code>	RunningStatus ...
<code>version</code>	<code>string</code>	

## `.status.base.appReleases`

Type

`array`

## `.status.base.appReleases[]`

Type

`string`

## API Endpoints

The following API endpoints are available:

- `/apis/cluster.alauda.io/v1alpha2/clustermodules`
  - `DELETE` : delete collection of ClusterModule
  - `GET` : list objects of kind ClusterModule
  - `POST` : create a new ClusterModule
- `/apis/cluster.alauda.io/v1alpha2/clustermodules/{name}`
  - `DELETE` : delete the specified ClusterModule
  - `GET` : read the specified ClusterModule
  - `PATCH` : partially update the specified ClusterModule
  - `PUT` : replace the specified ClusterModule
- `/apis/cluster.alauda.io/v1alpha2/clustermodules/{name}/status`
  - `GET` : read status of the specified ClusterModule
  - `PATCH` : partially update status of the specified ClusterModule
  - `PUT` : replace status of the specified ClusterModule

## `/apis/cluster.alauda.io/v1alpha2/clustermodules`

### HTTP method

`DELETE`

### Description

delete collection of ClusterModule

### HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
401 - Unauthorized	Empty

### HTTP method

`GET`

## Description

list objects of kind ClusterModule

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModuleList</code> schema
401 - Unauthorized	Empty

## HTTP method

POST

## Description

create a new ClusterModule

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate

Parameter	Type	Description
		fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

### Body parameters

Parameter	Type	Description
body	ClusterModule schema	application/json formatted

### HTTP responses

HTTP code	Response body
200 - OK	ClusterModule schema
201 - Created	ClusterModule schema
202 - Accepted	ClusterModule schema
401 - Unauthorized	Empty

## /apis/cluster.alauda.io/v1alpha2/clustermodules/{name}

### HTTP method

DELETE

### Description

delete the specified ClusterModule

### Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the

Parameter	Type	Description
		request. Valid values are: - All: all dry run stages will be processed

## HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

## HTTP method

GET

## Description

read the specified ClusterModule

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update the specified ClusterModule

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
401 - Unauthorized	Empty

## HTTP method

`PUT`

## Description

replace the specified ClusterModule

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ClusterModule</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
201 - Created	<code>ClusterModule</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

## /apis/cluster.alauda.io/v1alpha2/clustermodules/{name}/status

### HTTP method

GET

### Description

read status of the specified ClusterModule

### HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
401 - Unauthorized	Empty

### HTTP method

PATCH

### Description

partially update status of the specified ClusterModule

### Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing

Parameter	Type	Description
		unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace status of the specified ClusterModule

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ClusterModule</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterModule</code> schema
201 - Created	<code>ClusterModule</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

# CommonConfig

## [operator.alauda.io/v1alpha1]

### Description

CommonConfig is the Schema for the commonconfigs API

### Type

object

## Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources</a>
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		<a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds</a> ↗
metadata	ObjectMeta	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
spec	object	CommonConfigSpec defines the desired state of CommonConfig
status	object	CommonConfigStatus defines the observed state of CommonConfig

## .spec

### Description

CommonConfigSpec defines the desired state of CommonConfig

### Type

object

### Required

etcd global oidc release repositories

Property	Type	Description
elasticsearch	object	Elasticsearch Config

Property	Type	Description
<code>etcd</code>	<code>object</code>	Etcd Config
<code>ext</code>	<code>object</code>	Ext Config
<code>global</code>	<code>object</code>	Global configuration
<code>kafka</code>	<code>object</code>	Kafka Config
<code>oidc</code>	<code>object</code>	OIDC Config
<code>release</code>	<code>string</code>	Release verison
<code>repositories</code>	<code>array</code>	Repositories Config

## `.spec.elasticsearch`

### Description

Elasticsearch Config

### Type

`object`

Property	Type	Description
<code>host</code>	<code>string</code>	Server address
<code>nodes</code>	<code>array</code>	Log nodes
<code>password</code>	<code>string</code>	Password
<code>username</code>	<code>string</code>	Uername

## `.spec.elasticsearch.nodes`

### Description

Log nodes

### Type

`array`

## `.spec.elasticsearch.nodes[]`

### Type

`string`

## `.spec.etcd`

### Description

Etcd Config

### Type

`object`

**Required**

ectdCaSecret

ectdPeerSecret

servers

Property	Type	Description
ectdCaSecret	object	Etcd Ca Secret
ectdPeerSecret	object	Etcd Peer secret
servers	array	Etcd servers

**.spec.etcd.ectdCaSecret****Description**

Etcd Ca Secret

**Type**

object

**Required**

secretSource

Property	Type	Description
secretSource	object	

**.spec.etcd.ectdCaSecret.secretSource****Type**

object

**Required**

tls.crt

tls.key

Property	Type	Description
tls.crt	string	
tls.key	string	

## .spec.etcd.ectdPeerSecret

### Description

Etcd Peer secret

### Type

object

### Required

secretSource

Property	Type	Description
secretSource	object	

## .spec.etcd.ectdPeerSecret.secretSource

### Type

object

### Required

tls.crt

tls.key

Property	Type	Description
tls.crt	string	
tls.key	string	

## **.spec.etcd.servers**

### **Description**

Etcd servers

### **Type**

array

## **.spec.etcd.servers[]**

### **Type**

string

## **.spec.ext**

### **Description**

Ext Config

### **Type**

object

## **.spec.global**

### **Description**

Global configuration

### **Type**

object

### **Required**

apiAddress

defaultAdmin

erebusApiAddress

host

labelBaseDomain

replicas

tlsSecret

Property	Type	Description
<code>apiAddress</code>	<code>string</code>	Platform api address
<code>defaultAdmin</code>	<code>string</code>	The platform deploys a management account by default, and the default email <a href="mailto:admin@cpaas.io">admin@cpaas.io</a>
<code>erebusApiAddress</code>	<code>string</code>	Platform api gateway address
<code>host</code>	<code>string</code>	Platform access host address, support domain name or IP
<code>isOCP</code>	<code>boolean</code>	Whether to deploy on openshift cluster
<code>labelBaseDomain</code>	<code>string</code>	Platform resource instance label field, the default is <code>cpaas.io</code>
<code>namespace</code>	<code>string</code>	The namespace deployed by the platform, default <code>cpaas-system</code>
<code>replicas</code>	<code>integer</code>	The number of instances of platform common deployment components, the default is 2
<code>scheme</code>	<code>string</code>	Platform access protocol, support http or https

Property	Type	Description
<code>tlsSecret</code>	<code>object</code>	Tls Secret

## `.spec.global.tlsSecret`

### Description

Tls Secret

### Type

`object`

### Required

`secretSource`

Property	Type	Description
<code>secretSource</code>	<code>object</code>	

## `.spec.global.tlsSecret.secretSource`

### Type

`object`

### Required

`tls.crt`

`tls.key`

Property	Type	Description
<code>tls.crt</code>	<code>string</code>	
<code>tls.key</code>	<code>string</code>	

## `.spec.kafka`

## Description

Kafka Config

## Type

object

## Required

auth

host

kafka\_password

kafka\_user

zk\_acl\_password

zk\_password

zk\_user

Property	Type	Description
auth	boolean	Whether to enable authentication
host	string	Server address
kafka_password	string	
kafka_user	string	
zk_acl_password	string	
zk_password	string	
zk_user	string	

## .spec.oidc

## Description

OIDC Config

## Type

object

## Required

clientID

clientSecret

issuer

responseType

scopes

Property	Type	Description
clientID	string	OIDC Client ID
clientSecret	string	OIDC Client Secret
issuer	string	OIDC server address
responseType	string	OIDC Response Type
scopes	string	OIDC Scopes

## .spec.repositories

### Description

Repositories Config

### Type

array

## .spec.repositories[]

### Description

Repository Config

### Type

object

## Required

type url

Property	Type	Description
type	string	Repository type
url	string	Repository url

## .status

### Description

CommonConfigStatus defines the observed state of CommonConfig

### Type

object

## API Endpoints

The following API endpoints are available:

- `/apis/operator.alauda.io/v1alpha1/commonconfigs`
  - `DELETE` : delete collection of CommonConfig
  - `GET` : list objects of kind CommonConfig
  - `POST` : create a new CommonConfig
- `/apis/operator.alauda.io/v1alpha1/commonconfigs/{name}`
  - `DELETE` : delete the specified CommonConfig
  - `GET` : read the specified CommonConfig
  - `PATCH` : partially update the specified CommonConfig

- **PUT** : replace the specified CommonConfig
- `/apis/operator.alauda.io/v1alpha1/commonconfigs/{name}/status`
- **GET** : read status of the specified CommonConfig
- **PATCH** : partially update status of the specified CommonConfig
- **PUT** : replace status of the specified CommonConfig

## /apis/operator.alauda.io/v1alpha1/commonconfigs

### HTTP method

**DELETE**

### Description

delete collection of CommonConfig

### HTTP responses

HTTP code	Response body
200 - OK	<b>Status</b> schema
401 - Unauthorized	Empty

### HTTP method

**GET**

### Description

list objects of kind CommonConfig

### HTTP responses

HTTP code	Response body
200 - OK	<b>CommonConfigList</b> schema
401 - Unauthorized	Empty

### HTTP method

POST

## Description

create a new CommonConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>CommonConfig</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
201 - Created	<code>CommonConfig</code> schema
202 - Accepted	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

## /apis/operator.alauda.io/v1alpha1/commonconfigs/{name}

### HTTP method

DELETE

### Description

delete the specified CommonConfig

### Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

### HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

### HTTP method

GET

## Description

read the specified CommonConfig

## HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update the specified CommonConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace the specified CommonConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The

Parameter	Type	Description
		request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

### Body parameters

Parameter	Type	Description
body	CommonConfig schema	application/json formatted

### HTTP responses

HTTP code	Response body
200 - OK	CommonConfig schema
201 - Created	CommonConfig schema
401 - Unauthorized	Empty

## /apis/operator.alauda.io/v1alpha1/commonconfigs/{name}/status

### HTTP method

GET

### Description

read status of the specified CommonConfig

### HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update status of the specified CommonConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace status of the specified CommonConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<p><code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are:</p> <ul style="list-style-type: none"> <li>- Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23.</li> <li>- Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+.</li> <li>- Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.</li> </ul>

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>CommonConfig</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>CommonConfig</code> schema
201 - Created	<code>CommonConfig</code> schema
401 - Unauthorized	Empty

# ProductBase [product.alauda.io/v1alpha1]

## Description

ProductBase is the Schema for the productbases API

## Type

object

## Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	<p>APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources</a></p>

Property	Type	Description
kind	string	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds</a>
metadata	ObjectMeta	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
spec	object	ProductBaseSpec defines the desired state of ProductBase
status	object	ProductBaseStatus defines the observed state of ProductBase

## .spec

### Description

ProductBaseSpec defines the desired state of ProductBase

### Type

object

### Required

defaultAdmin

http

platformURL

registry

version

Property	Type	Description
<code>alternativeURLs</code>	<code>array</code>	AlternativeURLs is the platform's alternative urls
<code>defaultAdmin</code>	<code>object</code>	DefaultAdmin stands for the admin's config
<code>defaultRedirectPath</code>	<code>string</code>	DefaultRedirectPath stands for the default redirect path
<code>globalConfig</code>	<code>object</code>	GlobalConfig stands for the global config
<code>http</code>	<code>object</code>	HTTP stands for the http config
<code>ingress</code>	<code>object</code>	Ingress stands for the ingress configs
<code>platformPublicURL</code>	<code>string</code>	PlatformPublicURL stands the platform's public url
<code>platformURL</code>	<code>string</code>	PlatformURL is the platform's access url
<code>productBrand</code>	<code>string</code>	ProductBrand values: TKE、ACP、Common
<code>registry</code>	<code>object</code>	Registry is Docker Registry's Config

Property	Type	Description
<code>replicas</code>	<code>integer</code>	Replicas stands for the deploy replicas
<code>version</code>	<code>string</code>	Version stands for this product's version to deploy

## `.spec.alternativeURLs`

### Description

AlternativeURLs is the platform's alternative urls

### Type

`array`

## `.spec.alternativeURLs[]`

### Type

`string`

## `.spec.defaultAdmin`

### Description

DefaultAdmin stands for the admin's config

### Type

`object`

### Required

`account`

`email`

Property	Type	Description
<code>account</code>	<code>string</code>	Account stands for admin's account
<code>email</code>	<code>string</code>	Email stands for admin's email
<code>hash</code>	<code>string</code>	Hash stands for admin's password hash

## **.spec.globalConfig**

### **Description**

GlobalConfig stands for the global config

### **Type**

`object`

Property	Type	Description
<code>labels</code>	<code>object</code>	

## **.spec.globalConfig.labels**

### **Type**

`object`

## **.spec.http**

### **Description**

HTTP stands for the http config

### **Type**

object

**Required**

forceRedirectHttps

Property	Type	Description
forceRedirectHttps	boolean	ForceRedirectHTTPS stands for if force redirect to https
httpPort	integer	HTTPPort is the port of http
httpsPort	integer	HTTPPort is the port of https
tls	object	TLS stands for the server's TLS Certificate

**.spec.http.tls****Description**

TLS stands for the server's TLS Certificate

**Type**

object

Property	Type	Description
secretRef	object	SecretRef is the reference for tls secret

**.spec.http.tls.secretRef**

## Description

SecretRef is the reference for tls secret

## Type

object

## Required

name

Property	Type	Description
name	string	Name is the name of secret

## .spec.ingress

## Description

Ingress stands for the ingress configs

## Type

object

## Required

controller

host

ingressClassName

Property	Type	Description
annotations	object	
controller	object	Controller stands for the ingress controller configs
host	string	Host stands for the host config in ingress

Property	Type	Description
<code>ingressClassName</code>	<code>string</code>	IngressClassName stands for the ingressClassName of the ingress

## **.spec.ingress.annotations**

### Type

`object`

## **.spec.ingress.controller**

### Description

Controller stands for the ingress controller configs

### Type

`object`

### Required

`install`

Property	Type	Description
<code>install</code>	<code>boolean</code>	Install stands for if install ingress controller
<code>nodes</code>	<code>array</code>	Nodes stands for the ingress controller's running node

## **.spec.ingress.controller.nodes**

### Description

Nodes stands for the ingress controller's running node

## Type

array

## .spec.ingress.controller.nodes[]

## Type

string

## .spec.registry

## Description

Registry is Docker Registry's Config

## Type

object

## Required

address

preferPlatformURL

Property	Type	Description
address	string	Address is Docker Registry's address
external	boolean	External stands for a external registry
preferPlatformURL	boolean	PreferPlatformURL stands for business cluster prefer platform URL as registry address
secretRef	object	SecretRef is the reference for pullImageSecret

## .spec.registry.secretRef

### Description

SecretRef is the reference for pullImageSecret

### Type

object

### Required

name

Property	Type	Description
name	string	Name is the name of secret

## .status

### Description

ProductBaseStatus defines the observed state of ProductBase

### Type

object

### Required

phase

version

Property	Type	Description
artifacts	array	Artifacts stands for the product's artifacts
conditions	array	Conditions contains this product's conditions

Property	Type	Description
<code>globalID</code>	<code>string</code>	GlobalID stands for the global Id
<code>phase</code>	<code>string</code>	Phase stands for this product's phase
<code>replicas</code>	<code>integer</code>	Replicas stands for the product workload replicas
<code>version</code>	<code>string</code>	Version statnds for this product's current version

## **.status.artifacts**

### **Description**

Artifacts stands for the product's artifacts

### **Type**

`array`

## **.status.artifacts[]**

### **Description**

Artifact stands for the artifact's info

### **Type**

`object`

### **Required**

`channels`

`name`

`packageType`

Property	Type	Description
<code>channels</code>	<code>array</code>	Channels stands for the artifact's channels
<code>name</code>	<code>string</code>	Name stands for the artifact's name
<code>packageType</code>	<code>string</code>	PackageType stands for the artifact's package type

## `.status.artifacts[].channels`

### Description

Channels stands for the artifact's channels

### Type

`array`

## `.status.artifacts[].channels[]`

### Description

Channel stands for the artifact's channel

### Type

`object`

### Required

`artifactStatus`

`channel`

`default`

`displayName`

`repository`

`tag`

Property	Type	Description
<code>artifactStatus</code>	<code>string</code>	ArtifactStatus stands for the channel's artifact status
<code>brief</code>	<code>object</code>	Brief stands for the artifact's brief
<code>catalogSource</code>	<code>string</code>	CatalogSource stands for the catalog source to get the operator bundle
<code>categories</code>	<code>array</code>	Categories stands for the artifact's categories, such as: AI, Database, etc.
<code>channel</code>	<code>string</code>	Channel stands for the channel's name
<code>default</code>	<code>boolean</code>	Default stands for the artifact's default channel
<code>description</code>	<code>object</code>	Description stands for the channel's description
<code>digest</code>	<code>string</code>	Digest stands for the channel's digest

Property	Type	Description
<code>displayName</code>	<code>object</code>	DisplayName stands for the artifact's display name
<code>hidden</code>	<code>boolean</code>	Hidden stands for the artifact is hidden on the Marketplace's page
<code>packIgnore</code>	<code>boolean</code>	PackIgnore indicates whether to skip packaging
<code>provider</code>	<code>object</code>	Provider stands for the artifact's provider
<code>providerType</code>	<code>string</code>	ProviderType stands for the artifact's provider type, such as: platform, certified, custom, community
<code>repository</code>	<code>string</code>	Repository stands for the channel's repository
<code>supportedArchitectures</code>	<code>array</code>	SupportedArchitectures stands for the supported architectures, such as: amd64, arm64, etc.

Property	Type	Description
<code>supportedPlatformVersions</code>	<code>array</code>	SupportedPlatformVersions stands for the supported platform versions, such as: v4.0, v4.1, etc.
<code>supportedProtocolStacks</code>	<code>array</code>	SupportedProtocolStacks stands for the supported protocol stacks, such as: IPv4, IPv6, DualStack, etc.
<code>tag</code>	<code>string</code>	Tag stands for the channel's tag

## `.status.artifacts[].channels[].brief`

### Description

Brief stands for the artifact's brief

### Type

`object`

Property	Type	Description
<code>en</code>	<code>string</code>	EN stands for text with English
<code>zh</code>	<code>string</code>	ZH stands for text with Simple Chinese

## `.status.artifacts[].channels[].categories`

### Description

Categories stands for the artifact's categories, such as: AI, Database, etc.

### Type

array

## **.status.artifacts[].channels[].categories[]**

### Type

string

## **.status.artifacts[].channels[].description**

### Description

Description stands for the channel's description

### Type

object

Property	Type	Description
en	string	EN stands for text with English
zh	string	ZH stands for text with Simple Chinese

## **.status.artifacts[].channels[].displayName**

### Description

DisplayName stands for the artifact's display name

### Type

object

Property	Type	Description
en	string	EN stands for text with English
zh	string	ZH stands for text with Simple Chinese

## **.status.artifacts[].channels[].provider**

### Description

Provider stands for the artifact's provider

### Type

object

Property	Type	Description
en	string	EN stands for text with English
zh	string	ZH stands for text with Simple Chinese

## **.status.artifacts[].channels[].supportedArchitectures**

### Description

SupportedArchitectures stands for the supported architectures, such as: amd64, arm64, etc.

### Type

array

## **.status.artifacts[].channels[].supportedArchitectures[]**

## Type

string

## **.status.artifacts[].channels[].supportedPlatformVersions**

### Description

SupportedPlatformVersions stands for the supported platform versions, such as: v4.0, v4.1, etc.

## Type

array

## **.status.artifacts[].channels[].supportedPlatformVersions[]**

## Type

string

## **.status.artifacts[].channels[].supportedProtocolStacks**

### Description

SupportedProtocolStacks stands for the supported protocol stacks, such as: IPv4, IPv6, DualStack, etc.

## Type

array

## **.status.artifacts[].channels[].supportedProtocolStacks[]**

## Type

string

## **.status.conditions**

### Description

Conditions contains this product's conditions

## Type

array

## .status.conditions[]

## Description

ProductBaseCondition ...

## Type

object

## Required

status

type

Property	Type	Description
lastHeartbeatTime	string	LastHeartbeatTime stands for the timestamp to check Condition
lastTransitionTime	string	LastTransitionTime stands for the timestamp of Condition's status changed
message	string	Message stands for Condition's message
reason	string	Reason stands for Condition's reason
status	string	Status stands for Condition's status

Property	Type	Description
type	string	Type stands for Condition's type

## API Endpoints

The following API endpoints are available:

- `/apis/product.alauda.io/v1alpha2/productbases`
  - `DELETE` : delete collection of ProductBase
  - `GET` : list objects of kind ProductBase
  - `POST` : create a new ProductBase
- `/apis/product.alauda.io/v1alpha2/productbases/{name}`
  - `DELETE` : delete the specified ProductBase
  - `GET` : read the specified ProductBase
  - `PATCH` : partially update the specified ProductBase
  - `PUT` : replace the specified ProductBase
- `/apis/product.alauda.io/v1alpha2/productbases/{name}/status`
  - `GET` : read status of the specified ProductBase
  - `PATCH` : partially update status of the specified ProductBase
  - `PUT` : replace status of the specified ProductBase

### `/apis/product.alauda.io/v1alpha2/productbases`

#### HTTP method

`DELETE`

#### Description

delete collection of ProductBase

## HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
401 - Unauthorized	Empty

## HTTP method

GET

## Description

list objects of kind ProductBase

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBaseList</code> schema
401 - Unauthorized	Empty

## HTTP method

POST

## Description

create a new ProductBase

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing

Parameter	Type	Description
		<p>unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.</p>

## Body parameters

Parameter	Type	Description
body	ProductBase schema	application/json formatted

## HTTP responses

HTTP code	Response body
200 - OK	ProductBase schema
201 - Created	ProductBase schema
202 - Accepted	ProductBase schema
401 - Unauthorized	Empty

**/apis/product.alauda.io/v1alpha2/productbases/{name}**

**HTTP method**

**DELETE**

## Description

delete the specified ProductBase

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

## HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

## HTTP method

**GET**

## Description

read the specified ProductBase

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update the specified ProductBase

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace the specified ProductBase

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ProductBase</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
201 - Created	<code>ProductBase</code> schema
401 - Unauthorized	Empty

## /apis/product.alauda.io/v1alpha2/productbases/{name}/status

### HTTP method

GET

### Description

read status of the specified ProductBase

### HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
401 - Unauthorized	Empty

### HTTP method

PATCH

### Description

partially update status of the specified ProductBase

### Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
401 - Unauthorized	Empty

## HTTP method

`PUT`

## Description

replace status of the specified ProductBase

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ProductBase</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductBase</code> schema
201 - Created	<code>ProductBase</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

# ProductConfig [alauda.io/v1alpha1]

## Description

ProductConfig is the Schema for the productconfigs API

## Type

object

## Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	<p>APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources</a></p>

Property	Type	Description
kind	string	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds</a>
metadata	ObjectMeta	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
spec	object	ProductConfigSpec defines the desired state of ProductConfig
status	object	ProductConfigStatus defines the observed state of ProductConfig

## .spec

### Description

ProductConfigSpec defines the desired state of ProductConfig

### Type

object

### Required

availableVersion

brief

description

displayName

displayUIConfig

installUIConfig

logo

platformVersionMatch

productResource

type

Property	Type	Description
<code>availableVersion</code>	<code>string</code>	AvailableVersion stands for the available version to deploy
<code>brief</code>	<code>object</code>	Brief stands for the product's brief
<code>chartVersions</code>	<code>array</code>	ChartVersions stands for the charts when deploying, they are optional and will be added to the Prouct CR
<code>commonConfigRef</code>	<code>array</code>	CommonConfigRef stands for the reference from CommonConfig to Product CR
<code>description</code>	<code>object</code>	Description stands for the product's description
<code>displayName</code>	<code>object</code>	DisplayName stands for the product's display name
<code>displayUIConfig</code>	<code>object</code>	DisplayUIConfig stands for the web UI type when showing product's detail information
<code>installUIConfig</code>	<code>object</code>	InstallUIConfig stands for the web UI config when deploying product

Property	Type	Description
<code>isDisplay</code>	<code>boolean</code>	IsDisplay stands for if the product displaying on the products list
<code>logo</code>	<code>string</code>	Logo stands for the product logo with base64 encoded data uri
<code>platformVersionMatch</code>	<code>string</code>	PlatformVersionMatch stands for the product need match the platform version
<code>productResource</code>	<code>object</code>	ProductResource defines the target product CRD type
<code>singleApp</code>	<code>object</code>	SingleAppConfig stands for the config for SingleApp type
<code>subApp</code>	<code>object</code>	SubApp stands for the config for SubApp type
<code>type</code>	<code>string</code>	Type stands for the product's type
<code>upgradeUIConfig</code>	<code>array</code>	UpgradeUIConfig stands for the web UI config when upgrading product

Property	Type	Description
<code>upgradeableVersions</code>	<code>array</code>	UpgradeableVersions stands for the versions can upgrade from

## **.spec.brief**

### **Description**

Brief stands for the product's brief

### **Type**

`object`

Property	Type	Description
<code>en</code>	<code>string</code>	EN stands for text with English
<code>zh</code>	<code>string</code>	ZH stands for text with Simple Chinese

## **.spec.chartVersions**

### **Description**

ChartVersions stands for the charts when deploying, they are optional and will be added to the Prouct CR

### **Type**

`array`

## **.spec.chartVersions[]**

### **Description**

ChartVersion stands for chart version

### Type

object

### Required

name

version

Property	Type	Description
name	string	Name stands for the chart's name
version	string	Version stands for the chart's version

## .spec.commonConfigRef

### Description

CommonConfigRef stands for the reference from CommonConfig to Product CR

### Type

array

## .spec.commonConfigRef[]

### Description

KeysRef stands for key references with specified version

### Type

object

### Required

keys

versionMatch

Property	Type	Description
keys	array	Keys stands for a set of keys
versionMatch	string	VersionMatch stands for the target version

## .spec.commonConfigRef[].keys

### Description

Keys stands for a set of keys

### Type

array

## .spec.commonConfigRef[].keys[]

### Description

KeyRef stands for key reference

### Type

object

### Required

refKey

targetKey

Property	Type	Description
refKey	string	RefKey stands for source key
targetKey	string	TargetKey stands for target key

## .spec.description

### Description

Description stands for the product's description

### Type

object

Property	Type	Description
en	string	EN stands for text with English
zh	string	ZH stands for text with Simple Chinese

## .spec.displayName

### Description

DisplayName stands for the product's display name

### Type

object

Property	Type	Description
en	string	EN stands for text with English
zh	string	ZH stands for text with Simple Chinese

## .spec.displayUIConfig

## Description

DisplayUIConfig stands for the web UI type when showing product's detail information

## Type

object

## Required

type

Property	Type	Description
scheme	object	Scheme defines the dynamic form, used when Type is DynamicForm
type	string	Type defines the web UI type, [InUnderlord is DEPRECATED, use SubApp instead]

## .spec.displayUIConfig.scheme

## Description

Scheme defines the dynamic form, used when Type is DynamicForm

## Type

object

## .spec.installUIConfig

## Description

InstallUIConfig stands for the web UI config when deploying product

## Type

object

## Required

type

Property	Type	Description
scheme	object	Scheme defines the dynamic form, used when Type is DynamicForm
type	string	Type defines the web UI type, [InUnderlord is DEPRECATED, use SubApp instead]

## .spec.installUIConfig.scheme

### Description

Scheme defines the dynamic form, used when Type is DynamicForm

### Type

object

## .spec.productResource

### Description

ProductResource defines the target product CRD type

### Type

object

### Required

group

names

version

Property	Type	Description
group	string	Group stands for the target product's CRD Group

Property	Type	Description
<code>names</code>	<code>object</code>	Names stands for the target product's CRD's kind name
<code>version</code>	<code>string</code>	Version stands for the target product's CRD version

## `.spec.productResource.names`

### Description

Names stands for the target product's CRD's kind name

### Type

`object`

### Required

`kind`

`plural`

Property	Type	Description
<code>categories</code>	<code>array</code>	<code>categories</code> is a list of grouped resources this custom resource belongs to (e.g. 'all'). This is published in API discovery documents, and used by clients to support invocations like <code>kubectl get all</code> .
<code>kind</code>	<code>string</code>	<code>kind</code> is the serialized kind of the resource. It is normally CamelCase and singular. Custom resource instances will use this value as the <code>kind</code> attribute in API calls.

Property	Type	Description
<code>listKind</code>	<code>string</code>	<code>listKind</code> is the serialized kind of the list for this resource. Defaults to " <code>kind</code> List".
<code>plural</code>	<code>string</code>	<code>plural</code> is the plural name of the resource to serve. The custom resources are served under <code>/apis/&lt;group&gt;/&lt;version&gt;/.../&lt;plural&gt;</code> . Must match the name of the CustomResourceDefinition (in the form <code>&lt;names.plural&gt;.&lt;group&gt;</code> ). Must be all lowercase.
<code>shortNames</code>	<code>array</code>	<code>shortNames</code> are short names for the resource, exposed in API discovery documents, and used by clients to support invocations like <code>kubectl get &lt;shortname&gt;</code> . It must be all lowercase.
<code>singular</code>	<code>string</code>	<code>singular</code> is the singular name of the resource. It must be all lowercase. Defaults to lowercased <code>kind</code> .

## `.spec.productResource.names.categories`

### Description

`categories` is a list of grouped resources this custom resource belongs to (e.g. 'all'). This is published in API discovery documents, and used by clients to support invocations like ``kubectl get all``.

### Type

`array`

## `.spec.productResource.names.categories[]`

## Type

string

## .spec.productResource.names.shortNames

### Description

shortNames are short names for the resource, exposed in API discovery documents, and used by clients to support invocations like `kubectl get <shortname>`. It must be all lowercase.

## Type

array

## .spec.productResource.names.shortNames[]

## Type

string

## .spec.singleApp

### Description

SingleAppConfig stands for the config for SingleApp type

## Type

object

### Required

entrypoint

Property	Type	Description
entrypoint	string	Entrypoint stands for the entrypoint of standalone app

## **.spec.subApp**

### **Description**

SubApp stands for the config for SubApp type

### **Type**

object

### **Required**

entrypoint

name

Property	Type	Description
entrypoint	string	Entrypoint stands for the endpoint of sub app
name	string	Name stands for the name of sub app

## **.spec.upgradeUIConfig**

### **Description**

UpgradeUIConfig stands for the web UI config when upgrading product

### **Type**

array

## **.spec.upgradeUIConfig[]**

### **Description**

VersionedUIConfig stands for the web UI type with special version

### **Type**

object

### **Required**

type

versionMatch

Property	Type	Description
scheme	object	Scheme defines the dynamic form, used when Type is DynamicForm
type	string	Type defines the web UI type, [InUnderlord is DEPRECATED, use SubApp instead]
versionMatch	string	VersionMatch stands for the target version

## **.spec.upgradeUIConfig[].scheme**

### **Description**

Scheme defines the dynamic form, used when Type is DynamicForm

### **Type**

object

## **.spec.upgradeableVersions**

### **Description**

UpgradeableVersions stands for the versions can upgrade from

### **Type**

array

## **.spec.upgradeableVersions[]**

### **Type**

`string`

## .status

### Description

ProductConfigStatus defines the observed state of ProductConfig

### Type

`object`

### Required

`message``phase``reason`

Property	Type	Description
<code>message</code>	<code>string</code>	Message contains the detail messages about the status
<code>phase</code>	<code>string</code>	Phase stands for the ProductConfig's status
<code>reason</code>	<code>string</code>	Reason explain why the ProductCongfig in this status

## API Endpoints

The following API endpoints are available:

- `/apis/alauda.io/v1alpha1/productconfigs`
  - `DELETE` : delete collection of ProductConfig
  - `GET` : list objects of kind ProductConfig
  - `POST` : create a new ProductConfig
- `/apis/alauda.io/v1alpha1/productconfigs/{name}`

- **DELETE** : delete the specified ProductConfig
- **GET** : read the specified ProductConfig
- **PATCH** : partially update the specified ProductConfig
- **PUT** : replace the specified ProductConfig
- `/apis/alauda.io/v1alpha1/productconfigs/{name}/status`
  - **GET** : read status of the specified ProductConfig
  - **PATCH** : partially update status of the specified ProductConfig
  - **PUT** : replace status of the specified ProductConfig

## /apis/alauda.io/v1alpha1/productconfigs

### HTTP method

**DELETE**

### Description

delete collection of ProductConfig

### HTTP responses

HTTP code	Response body
200 - OK	<b>Status</b> schema
401 - Unauthorized	Empty

### HTTP method

**GET**

### Description

list objects of kind ProductConfig

### HTTP responses

HTTP code	Response body
200 - OK	<b>ProductConfigList</b> schema

HTTP code	Response body
401 - Unauthorized	Empty

## HTTP method

POST

## Description

create a new ProductConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
body	ProductConfig schema	application/json formatted

## HTTP responses

HTTP code	Response body
200 - OK	ProductConfig schema
201 - Created	ProductConfig schema
202 - Accepted	ProductConfig schema
401 - Unauthorized	Empty

## /apis/alauda.io/v1alpha1/productconfigs/{name}

### HTTP method

DELETE

### Description

delete the specified ProductConfig

### Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

## HTTP responses

HTTP code	Response body
200 - OK	Status schema

HTTP code	Response body
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

## HTTP method

GET

## Description

read the specified ProductConfig

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update the specified ProductConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace the specified ProductConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing

Parameter	Type	Description
		unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

### Body parameters

Parameter	Type	Description
body	ProductConfig schema	application/json formatted

### HTTP responses

HTTP code	Response body
200 - OK	ProductConfig schema
201 - Created	ProductConfig schema
401 - Unauthorized	Empty

## /apis/alauda.io/v1alpha1/productconfigs/{name}/status

### HTTP method

GET

## Description

read status of the specified ProductConfig

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update status of the specified ProductConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate

Parameter	Type	Description
		fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductConfig</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace status of the specified ProductConfig

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields.

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
body	ProductConfig schema	application/json formatted

## HTTP responses

HTTP code	Response body
200 - OK	ProductConfig schema
201 - Created	ProductConfig schema
401 - Unauthorized	Empty

# ProductUpgrade

## [product.alauda.io/v1alpha1]

### Description

ProductUpgrade is the Schema for the productupgrades API

### Type

object

## Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: <a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources</a>
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		<a href="https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds">https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds</a> ↗
metadata	ObjectMeta	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
spec	object	ProductUpgradeSpec defines the desired state of ProductUpgrade
status	object	ProductUpgradeStatus defines the observed state of ProductUpgrade

## .spec

### Description

ProductUpgradeSpec defines the desired state of ProductUpgrade

### Type

object

### Required

onlineUpgrade

Property	Type	Description
onlineUpgrade	object	OnlineUpgrade is the online upgrade configuration

Property	Type	Description
pause	boolean	Pause is a flag to pause the upgrade
targetArch	string	TargetArch is the architecture of the product to upgrade to
targetVersion	string	TargetVersion is the version of the product to upgrade to
type	string	Type is the type of product upgrade
upgrade	boolean	Upgrade is a flag of to do the upgrade

## .spec.onlineUpgrade

### Description

OnlineUpgrade is the online upgrade configuration

### Type

object

### Required

checkFrequency

enable

enableMsgNotify

Property	Type	Description
checkFrequency	integer	CheckFrequency is the frequency to check for new version

Property	Type	Description
enable	boolean	Enable is a flag to enable online upgrade
enableMsgNotify	boolean	EnableMsgNotify is a flag to enable message notification

## .status

### Description

ProductUpgradeStatus defines the observed state of ProductUpgrade

### Type

object

### Required

phase source status

Property	Type	Description
lastCheckTimestamp	string	LastCheckTimestamp is the last time the product upgrade was checked
message	string	Message is the message of the product upgrade
phase	string	Phase is the phase of the product upgrade

Property	Type	Description
<code>preheatVersions</code>	<code>array</code>	PreHeatVersions is the pre-heat histories of the product upgrade
<code>source</code>	<code>object</code>	OnlineUpgradeSource is the source of online upgrade
<code>status</code>	<code>string</code>	Status is the status of the product upgrade
<code>upgradable</code>	<code>boolean</code>	Upgradable is a flag to indicate whether the product is upgradable
<code>upgradeBlockReason</code>	<code>string</code>	UpgradeBlockReason is the reason why the product is not upgradable
<code>upgradePaths</code>	<code>array</code>	UpgradePaths is the upgrade paths of the product upgrade

## **.status.preheatVersions**

### **Description**

PreHeatVersions is the pre-heat histories of the product upgrade

### **Type**

`array`

# .status.preheatVersions[]

## Description

PreheatHistory defines the pre-heat history

## Type

object

## Required

arch

status

version

Property	Type	Description
arch	string	Arch is the architecture of the new version
artifactsAll	integer	ArtifactsAll is the number of artifacts
artifactsSynced	integer	ArtifactsSynced is the number of artifacts synced
baseOperatorVer	string	BaseOperatorVer is the version of the base operator
preHeatEndTime	string	PreHeatEndTime is the end time of pre-heat
preHeatStartTime	string	PreHeatStartTime is the start time of pre-heat
progress	integer	Progress is the progress of pre-heat

Property	Type	Description
<code>status</code>	<code>string</code>	PreHeatStatus is the pre-heat status
<code>version</code>	<code>string</code>	Version is the version of the product to upgrade to

## .status.source

### Description

OnlineUpgradeSource is the source of online upgrade

### Type

`object`

### Required

`registry`

Property	Type	Description
<code>envId</code>	<code>integer</code>	EnvId is the environment id
<code>insecure</code>	<code>boolean</code>	Insecure is a flag to enable insecure registry
<code>plainHttp</code>	<code>boolean</code>	PlainHttp is a flag to enable plain http
<code>registry</code>	<code>string</code>	Registry is the cloud registry

Property	Type	Description
<code>tenantId</code>	<code>integer</code>	TenantId is the tenant id

## `.status.upgradePaths`

### Description

UpgradePaths is the upgrade paths of the product upgrade

### Type

`array`

## `.status.upgradePaths[]`

### Description

UpgradePath defines the upgrade path

### Type

`object`

### Required

`arches`

`release`

`sizes`

`version`

Property	Type	Description
<code>arches</code>	<code>array</code>	Arches is the architectures of the new version
<code>release</code>	<code>string</code>	Release is the release version of the product to upgrade to
<code>sizes</code>	<code>array</code>	Sizes is the architectures of the new version

Property	Type	Description
<code>tags</code>	<code>array</code>	Tags is the tags of the new version
<code>version</code>	<code>string</code>	Version is the version of the product to upgrade to

## `.status.upgradePaths[].arches`

### Description

Arches is the architectures of the new version

### Type

`array`

## `.status.upgradePaths[].arches[]`

### Type

`string`

## `.status.upgradePaths[].sizes`

### Description

Sizes is the architectures of the new version

### Type

`array`

## `.status.upgradePaths[].sizes[]`

### Type

`integer`

## `.status.upgradePaths[].tags`

### Description

Tags is the tags of the new version

### Type

array

## `.status.upgradePaths[].tags[]`

### Type

string

## API Endpoints

The following API endpoints are available:

- `/apis/product.alauda.io/v1alpha1/productupgrades`
  - `DELETE` : delete collection of ProductUpgrade
  - `GET` : list objects of kind ProductUpgrade
  - `POST` : create a new ProductUpgrade
- `/apis/product.alauda.io/v1alpha1/productupgrades/{name}`
  - `DELETE` : delete the specified ProductUpgrade
  - `GET` : read the specified ProductUpgrade
  - `PATCH` : partially update the specified ProductUpgrade
  - `PUT` : replace the specified ProductUpgrade
- `/apis/product.alauda.io/v1alpha1/productupgrades/{name}/status`
  - `GET` : read status of the specified ProductUpgrade
  - `PATCH` : partially update status of the specified ProductUpgrade
  - `PUT` : replace status of the specified ProductUpgrade

# /apis/product.alauda.io/v1alpha1/productupgrades

## HTTP method

DELETE

## Description

delete collection of ProductUpgrade

## HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
401 - Unauthorized	Empty

## HTTP method

GET

## Description

list objects of kind ProductUpgrade

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgradeList</code> schema
401 - Unauthorized	Empty

## HTTP method

POST

## Description

create a new ProductUpgrade

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ProductUpgrade</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
201 - Created	<code>ProductUpgrade</code> schema

HTTP code	Response body
202 - Accepted	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

## /apis/product.alauda.io/v1alpha1/productupgrades/{name}

### HTTP method

DELETE

### Description

delete the specified ProductUpgrade

### Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

### HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

### HTTP method

GET

### Description

read the specified ProductUpgrade

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

## HTTP method

`PATCH`

## Description

partially update the specified ProductUpgrade

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<p><code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are:</p> <ul style="list-style-type: none"> <li>- Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23.</li> <li>- Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+.</li> <li>- Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.</li> </ul>

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

## HTTP method

`PUT`

## Description

replace the specified `ProductUpgrade`

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code>

Parameter	Type	Description
		directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## Body parameters

Parameter	Type	Description
<code>body</code>	<code>ProductUpgrade</code> schema	<code>application/json</code> formatted

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
201 - Created	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

# /apis/product.alauda.io/v1.alpha1/productupgrades/{name}/status

## HTTP method

GET

## Description

read status of the specified ProductUpgrade

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

## HTTP method

PATCH

## Description

partially update status of the specified ProductUpgrade

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a

Parameter	Type	Description
		warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

## HTTP responses

HTTP code	Response body
200 - OK	<code>ProductUpgrade</code> schema
401 - Unauthorized	Empty

## HTTP method

PUT

## Description

replace status of the specified ProductUpgrade

## Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently

Parameter	Type	Description
		<p>dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.</p>

## Body parameters

Parameter	Type	Description
body	ProductUpgrade schema	application/json formatted

## HTTP responses

HTTP code	Response body
200 - OK	ProductUpgrade schema
201 - Created	ProductUpgrade schema
401 - Unauthorized	Empty