Platform Base APIs

ClusterModule [cluster.alauda.i CommonConfig [operator.alaud ProductBas

ProductConfig [alauda.io/v1alp ProductUpgrade [product.alauda.io/v1alpha1]

ClusterModule [cluster.alauda.io/v1alpha1]

Description

ClusterModule is the Schema for the clustermodules API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ClusterModuleSpec defines the desired state of ClusterModule
`status`	`object`	ClusterModuleStatus defines the observed state of ClusterModule

.spec

Description

ClusterModuleSpec defines the desired state of ClusterModule

Type

`object`

Property	Type	Description
`parallelUpgrade`	`boolean`	ParallelUpgrade stands for if the upgrading could execute parallelly.
`upgradeModulesInfos`	`array`	UpgradeModuleInfos stands for the information to upgrade the modules
`valuesOverride`	`object`	ValuesOverride ...
`version`	`string`	

.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
`base`	`object`	ModuleStatus ...
`editable`	`boolean`	
`needUpgrade`	`boolean`	
`sync`	`string`	SyncMethod ...
`unableUpgradeReason`	`string`	
`upgradable`	`boolean`	

.status.appReleases

Type

`object`

.status.base

Description

ModuleStatus ...

Type

`object`

Required

`deployStatus` `runningStatus`

Property	Type	Description
`appReleases`	`array`	

Property	Type	Description
`deployStatus`	`string`	DeployStatus ...
`deployTime`	`string`	
`lastProbeTime`	`string`	
`lastTransitionTime`	`string`	
`logPath`	`string`	
`message`	`string`	
`reason`	`string`	
`runningStatus`	`string`	RunningStatus ...
`version`	`string`	

.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	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ClusterModule

HTTP responses

HTTP code	Response body
200 - OK	`ClusterModuleList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new 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 request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	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	`Status` schema
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ClusterModule

HTTP responses

HTTP code	Response body
200 - OK	`ClusterModule` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ClusterModule

Query parameters

Parameter	Type	Description
`dryRun`	`string`	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
`fieldValidation`	`string`	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	`ClusterModule` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace 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 request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	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
`body`	`ClusterModule` schema	`application/json` formatted

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of 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 request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	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	`ClusterModule` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of 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 request. Valid values are: - All: all dry run stages will be processed
`fieldValidation`	`string`	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
`body`	`ClusterModule` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ClusterModule` schema
201 - Created	`ClusterModule` 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
`apiVersion`	`string`	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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`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:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`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
`etcd`	`object`	Etcd Config
`ext`	`object`	Ext Config
`global`	`object`	Global configuration
`kafka`	`object`	Kafka Config
`oidc`	`object`	OIDC Config
`release`	`string`	Release verison
`repositories`	`array`	Repositories Config

.spec.elasticsearch

Description

Elasticsearch Config

Type

`object`

Property	Type	Description
`host`	`string`	Server address
`nodes`	`array`	Log nodes
`password`	`string`	Password
`username`	`string`	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
`apiAddress`	`string`	Platform api address
`defaultAdmin`	`string`	The platform deploys a management account by default, and the default email admin@cpaas.io ↗
`erebusApiAddress`	`string`	Platform api gateway address
`host`	`string`	Platform access host address, support domain name or IP
`isOCP`	`boolean`	Whether to deploy on openshift cluster
`labelBaseDomain`	`string`	Platform resource instance label field, the default is cpaas.io
`namespace`	`string`	The namespace deployed by the platform, default cpaas-system
`replicas`	`integer`	The number of instances of platform common deployment components, the default is 2
`scheme`	`string`	Platform access protocol, support http or https

Property	Type	Description
`tlsSecret`	`object`	Tls Secret

.spec.global.tlsSecret

Description

Tls Secret

Type

`object`

Required

`secretSource`

Property	Type	Description
`secretSource`	`object`	

.spec.global.tlsSecret.secretSource

Type

`object`

Required

`tls.crt` `tls.key`

Property	Type	Description
`tls.crt`	`string`	
`tls.key`	`string`	

.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

HTTP method

`DELETE`

Description

delete collection of CommonConfig

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind CommonConfig

HTTP responses

HTTP code	Response body
200 - OK	`CommonConfigList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new CommonConfig

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
`fieldValidation`	`string`	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
`body`	`CommonConfig` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`CommonConfig` schema
201 - Created	`CommonConfig` schema

HTTP code	Response body
202 - Accepted	`CommonConfig` schema
401 - Unauthorized	Empty

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

HTTP method

`DELETE`

Description

delete the specified CommonConfig

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
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified CommonConfig

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified CommonConfig

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

Parameter	Type	Description
`fieldValidation`	`string`	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	`CommonConfig` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified CommonConfig

Query parameters

Parameter	Type	Description
`dryRun`	`string`	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
`fieldValidation`	`string`	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
`body`	`CommonConfig` schema	`application/json` formatted

HTTP responses

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

ProductBase [product.alauda.io/v1alpha1]

Description

ProductBase is the Schema for the productbases API

Type

`object`

Specification

Property	Type	Description
`apiVersion`	`string`	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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`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
`alternativeURLs`	`array`	AlternativeURLs is the platform's alternative urls
`defaultAdmin`	`object`	DefaultAdmin stands for the admin's config
`defaultRedirectPath`	`string`	DefaultRedirectPath stands for the default redirect path
`globalConfig`	`object`	GlobalConfig stands for the global config
`http`	`object`	HTTP stands for the http config
`ingress`	`object`	Ingress stands for the ingress configs
`platformPublicURL`	`string`	PlatformPublicURL stands the platform's public url
`platformURL`	`string`	PlatformURL is the platform's access url
`productBrand`	`string`	ProductBrand values: TKE、ACP、Common
`registry`	`object`	Registry is Docker Registry's Config

Property	Type	Description
`replicas`	`integer`	Replicas stands for the deploy replicas
`version`	`string`	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
`account`	`string`	Account stands for admin's account
`email`	`string`	Email stands for admin's email
`hash`	`string`	Hash stands for admin's password hash

.spec.globalConfig

Description

GlobalConfig stands for the global config

Type

`object`

Property	Type	Description
`labels`	`object`	

.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 statnds for the server's TLS Certificate

.spec.http.tls

Description

TLS statnds 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
`ingressClassName`	`string`	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
`install`	`boolean`	Install stands for if install ingress controller
`nodes`	`array`	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
`globalID`	`string`	GlobalID stands for the global Id
`phase`	`string`	Phase stands for this product's phase
`replicas`	`integer`	Replicas stands for the product workload replicas
`version`	`string`	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
`channels`	`array`	Channels stands for the artifact's channels
`name`	`string`	Name stands for the artifact's name
`packageType`	`string`	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
`artifactStatus`	`string`	ArtifactStatus stands for the channel's artifact status
`brief`	`object`	Brief stands for the artifact's brief
`catalogSource`	`string`	CatalogSource stands for the catalog source to get the operator bundle
`categories`	`array`	Categories stands for the artifact's categories, such as: AI, Database, etc.
`channel`	`string`	Channel stands for the channel's name
`default`	`boolean`	Default stands for the artifact's default channel
`description`	`object`	Description stands for the channel's description
`digest`	`string`	Digest stands for the channel's digest

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

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

.status.artifacts[].channels[].brief

Description

Brief stands for the artifact's brief

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[].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	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ProductBase

HTTP responses

HTTP code	Response body
200 - OK	`ProductBaseList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new ProductBase

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
`fieldValidation`	`string`	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`	`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
`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
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ProductBase

HTTP responses

HTTP code	Response body
200 - OK	`ProductBase` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ProductBase

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
`fieldValidation`	`string`	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	`ProductBase` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ProductBase

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
`fieldValidation`	`string`	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
`body`	`ProductBase` schema	`application/json` formatted

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ProductBase

Query parameters

Parameter	Type	Description
`dryRun`	`string`	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
`fieldValidation`	`string`	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	`ProductBase` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ProductBase

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
`fieldValidation`	`string`	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
`body`	`ProductBase` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ProductBase` schema
201 - Created	`ProductBase` 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
`apiVersion`	`string`	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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`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
`availableVersion`	`string`	AvailableVersion stands for the available version to deploy
`brief`	`object`	Brief stands for the product's brief
`chartVersions`	`array`	ChartVersions stands for the charts when deploying, they are optional and will be added to the Prouct CR
`commonConfigRef`	`array`	CommonConfigRef stands for the reference from CommonConfig to Product CR
`description`	`object`	Description stands for the product's description
`displayName`	`object`	DisplayName stands for the product's display name
`displayUIConfig`	`object`	DisplayUIConfig stands for the web UI type when showing product's detail information
`installUIConfig`	`object`	InstallUIConfig stands for the web UI config when deploying product

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

Property	Type	Description
`upgradeableVersions`	`array`	UpgradeableVersions stands for the versions can upgrade from

.spec.brief

Description

Brief stands for the product's brief

Type

`object`

Property	Type	Description
`en`	`string`	EN stands for text with English
`zh`	`string`	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
`names`	`object`	Names stands for the target product's CRD's kind name
`version`	`string`	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
`categories`	`array`	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`.
`kind`	`string`	kind is the serialized kind of the resource. It is normally CamelCase and singular. Custom resource instances will use this value as the `kind` attribute in API calls.

Property	Type	Description
`listKind`	`string`	listKind is the serialized kind of the list for this resource. Defaults to "`kind`List".
`plural`	`string`	plural is the plural name of the resource to serve. The custom resources are served under `/apis/<group>/<version>/.../<plural>`. Must match the name of the CustomResourceDefinition (in the form `<names.plural>.<group>`). Must be all lowercase.
`shortNames`	`array`	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.
`singular`	`string`	singular is the singular name of the resource. It must be all lowercase. Defaults to lowercased `kind`.

.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 SignleApp 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 entrypoint 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
`message`	`string`	Message contains the detail messages about the status
`phase`	`string`	Phase stands for the ProductConfig's status
`reason`	`string`	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	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ProductConfig

HTTP responses

HTTP code	Response body
200 - OK	`ProductConfigList` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new 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
`fieldValidation`	`string`	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
`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	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ProductConfig

HTTP responses

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

HTTP method

`PATCH`

Description

partially update 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
`fieldValidation`	`string`	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	`ProductConfig` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace 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
`fieldValidation`	`string`	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	`ProductConfig` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of 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
`fieldValidation`	`string`	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	`ProductConfig` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of 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
`fieldValidation`	`string`	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.

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
`apiVersion`	`string`	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: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
`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:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`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
`preheatVersions`	`array`	PreHeatVersions is the pre-heat histories of the product upgrade
`source`	`object`	OnlineUpgradeSource is the source of online upgrade
`status`	`string`	Status is the status of the product upgrade
`upgradable`	`boolean`	Upgradable is a flag to indicate whether the product is upgradable
`upgradeBlockReason`	`string`	UpgradeBlockReason is the reason why the product is not upgradable
`upgradePaths`	`array`	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
`status`	`string`	PreHeatStatus is the pre-heat status
`version`	`string`	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
`envId`	`integer`	EnvId is the environment id
`insecure`	`boolean`	Insecure is a flag to enable insecure registry
`plainHttp`	`boolean`	PlainHttp is a flag to enable plain http
`registry`	`string`	Registry is the cloud registry

Property	Type	Description
`tenantId`	`integer`	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
`arches`	`array`	Arches is the architectures of the new version
`release`	`string`	Release is the release version of the product to upgrade to
`sizes`	`array`	Sizes is the architectures of the new version

Property	Type	Description
`tags`	`array`	Tags is the tags of the new version
`version`	`string`	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	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind ProductUpgrade

HTTP responses

HTTP code	Response body
200 - OK	`ProductUpgradeList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new ProductUpgrade

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
`fieldValidation`	`string`	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
`body`	`ProductUpgrade` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ProductUpgrade` schema
201 - Created	`ProductUpgrade` schema

HTTP code	Response body
202 - Accepted	`ProductUpgrade` schema
401 - Unauthorized	Empty

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

HTTP method

`DELETE`

Description

delete the specified ProductUpgrade

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
202 - Accepted	`Status` schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ProductUpgrade

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified ProductUpgrade

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

Parameter	Type	Description
`fieldValidation`	`string`	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	`ProductUpgrade` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ProductUpgrade

Query parameters

Parameter	Type	Description
`dryRun`	`string`	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
`fieldValidation`	`string`	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
`body`	`ProductUpgrade` schema	`application/json` formatted

HTTP responses

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

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

HTTP method

`GET`

Description

read status of the specified ProductUpgrade

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ProductUpgrade

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
`fieldValidation`	`string`	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

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	`ProductUpgrade` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ProductUpgrade

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
`fieldValidation`	`string`	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
		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`	`ProductUpgrade` schema	`application/json` formatted

HTTP responses

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