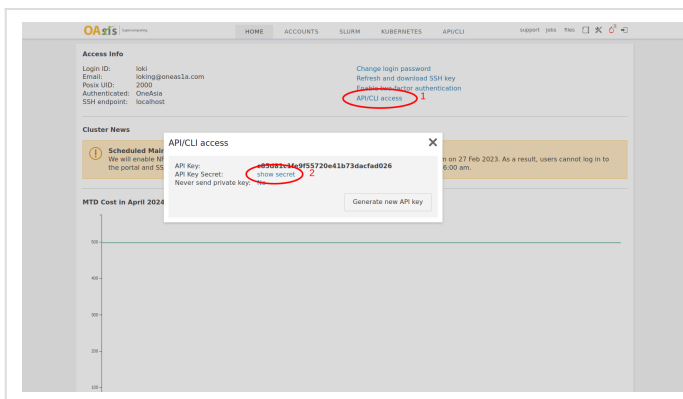


Onboard a new tenant demo with API.

In this demonstration, we will show you how to use the API to create a new billing account with a suitable quota configuration. We will first create a new account, then a new user, and grant the user cooperation permissions. Finally, we'll assign the new account quota to our Slurm and k8s cluster.



First, we need to retrieve our token for API access. Go to the dashboard and obtain your own API access key and secret.

There are various ways to store the API key and secret. In the following demo, we will use the shell command to export it as a shell environment variable. However, this method is not secure for storing API access. If you have any security concerns, please generate a new API key.

```
export OASIS_KEY=<API KEY>
export OASIS_SECRET=<API KEY SECRET>
export OASIS_HOST=<API DOMAIN>
```

All the preparations are complete; let's begin. First, we need to understand what billing accounts currently exist. Considering that the billing account structure is hierarchical, the root billing account is 'ALL'.

```
#get parentOpts

curl -k -XGET https://$OASIS_DOMAIN/admin/accounts/new \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XGET https://$OASIS_DOMAIN/admin/accounts/new \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
{"ok":1,"parentOpts":["oneasia","all"],"query":{}}%
```

Now, we can create a new billing account. For example, let's say we want to create an account under OneAsia, owned by 'Test'. This account can have a maximum of 10 users, with a unique GID of 4001.

```
# create a new billing account

curl -k -XPOST https://$OASIS_DOMAIN/admin/accounts/new \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d
'{"name":"test","parent":"oneasia","type":"billing","owner_name":"test","owner_email":"test@test.com",
"fairshare":100,"user_quota":10,"gid":4001}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/admin/accounts/new \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"name":"test","parent":"oneasia","type":"billing","owner_name":"test","owner_email":"test@test.com",
"fairshare":100,"user_quota":10,"gid":4001}'
{"ok":1}%
```

After that, we can create a consumer account under the newly created billing account. The consumer account is the actual account that the researcher will be working on.

```
# create a consumer account
curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/oneasiaai/child-accounts/new-account \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d
'{"gid":1031,"fairshare":100,"user_quota":10,"parent":"test","type":"consumer","owner_email":"test2@test.com","name":"test_child"}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/oneasiaai/child-accounts/new-account \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"gid":1031,"fairshare":100,"user_quota":10,"parent":"test","type":"consumer","owner_email":"test2@test.com","name":"test_child"}'
{"ok":1}%
```

Then, we can create a user

```
# create a new user
curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/account/test/create-user \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"login_id":"test","full_name":"test test","email":"test@test.com","password":"password12345678","mobile":91235678}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/account/test/create-user \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"login_id":"test","full_name":"test test","email":"test@test.com","password":"password12345678","mobile":91235678}'
{"ok":1}%
```


We need to assign coordinator permissions to the manager's account. The account coordinator has the authority to modify the configuration of that account and its subordinate accounts.

```
# grant coordinator permission to that account
curl -k -XPOST https://$OASIS_DOMAIN/admin/accounts/test/user/test/set-coordinator \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":true}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/admin/accounts/test/user/test/set-coordinator \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":true}'
{"ok":1}
```

Finally, we onboard a new billing account, facilitated by a coordinator.

 Supercomputing

HOME





ACCOUNTS

SLURM

KUBERNETES

API/CLI

support jobs files

Accounts > test

Overview

Child accounts

Usage: Job

Usage: Storage

Usage: Efficiency

Billing

Settings

Subscriptions

Users 1/10

1 - 1 of 1 < Prev Next > ⚙

Id	Email	Is Coordinator	Actions
test	test@test.com	Yes	revoke access

Jobs

[View running jobs](#)

[View completed jobs](#)

Invite an existing user by email

Create and add a new user

The next section is assign a proper quota in slurm. In the web interface, it is in the setting section.

Overview

Child accounts

Usage: Job

Usage: Storage

Usage: Efficiency

Billing

Settings

Subscriptions

Usage Quota

Quota Editable by Coordinator

Reset period

CPU Builtin

GPU Builtin

CPU Shared

GPU Shared

CPU Test

GPU Test

Behavior when quota exceeded

True

None

next: N/A

Unlimited (Usage:0)

Unlimited (Usage:0)

Unlimited (Usage:0)

Unlimited (Usage:0)

Unlimited (Usage:0)

Unlimited (Usage:0)

Notify Only

Budget Control

Budget Editable by Coordinator

Reset period

Budget (HKD)

Notify when budget consumed over

Restrict when budget consumed over

True

Monthly

next: 2024.05

Unlimited, MTD Charge: HKD 0

No notification

No restriction

* Budget quota changes may take 60 minutes to apply.

Storage Class max storage for each K8s cluster

Storage quota Editable by Coordinator

Cluster: dev

False

0 GB for storage class default

0 GB for storage class standard

GPU Quota for each K8s cluster

GPU Quota Editable by Coordinator

Cluster: dev

False

Guaranteed 0 card

Max unlimited card

Account Management

Status

Active

First, we can disable the setting that allows the coordinator to edit the quota.

```
#change slurm quota editable from coordinator
curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota-editable-by-coordinator \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":"No"}'
```

The result should be like the following:

```

don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota-editable-by-coordinator \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":"No"}'
{"ok":1}%

```

Next, we can set the Slurm quota by resource pool. This example assumes that the Slurm cluster has two resource pools: 'builtin' and 'shared'.

#set slurm quota to that account, for example if cluster has two resource pool: builtin and shared

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/builtin_csu \
```

```
-H "Content-Type: application/json" \
```

```
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
```

```
-d '{"value":50}'
```

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/builtin_gsu \
```

```
-H "Content-Type: application/json" \
```

```
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
```

```
-d '{"value":50}'
```

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/shared_csu \
```

```
-H "Content-Type: application/json" \
```

```
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
```

```
-d '{"value":50}'
```

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/shared_gsu \
```

```
-H "Content-Type: application/json" \
```

```
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
```

```
-d '{"value":50}'
```

The result should be like the following:

```

don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/builtin_csu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}%
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/builtin_gsu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}%
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/shared_csu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}%
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-quota/shared_gsu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}%

```

The default action when a quota is exceeded is inaction. However, we can modify this to either notify us or automatically terminate jobs.

```
#change the slurm quota exceeded action: option include: auto_kill_jobs, no_action,notify_only
curl -XPOST https://$OASIS_DOMAIN/my-accounts/oneasiaai/settings/update/quota-exceeded-action \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":"auto_kill_jobs"}
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/oneasiaai/settings/update/quota-exceeded-action \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":"auto_kill_jobs"}'
{"ok":1}
```

Next, we can adjust the budget settings. Both quota and budget are limitations on team usage. The quota is from a resource perspective, while the budget is from a financial perspective.

```
#change slurm budget editable from coordinator
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-budget-editable-by-coordinator \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":"No"}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-budget-editable-by-coordinator \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":"No"}'
{"ok":1}
```

In this demo, we set 20000 as budget

```
#set money-wise budget to that account
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-budget \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":20000}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/set-budget \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":20000}'
{"ok":1}
```

In a Slurm budget, you can set notifications and job launch restrictions to activate when an account exceeds 50% of its budget.

```
#set notification when budget consumed over certain ratio
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/settings/update/budget-exceeded-ratio/notify \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":50}'

# set job restriction when budget consumed over certain ratio, which take 60 minutes to apply
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/settings/update/budget-exceeded-ratio/restrict \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":50}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/settings/update/budget-exceeded-ratio/notify \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/settings/update/budget-exceeded-ratio/restrict \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value":50}'
{"ok":1}
```

We finish config slurm quota and budget. you can see the change is already show in web UI.

Usage Quota

Quota Editable by Coordinator	False
Reset period	None
	next: N/A
CPU Builtin	0 / 50 (0.0%)
GPU Builtin	0 / 50 (0.0%)
CPU Shared	0 / 50 (0.0%)
GPU Shared	0 / 50 (0.0%)
CPU Test	Unlimited (Usage:0)
GPU Test	Unlimited (Usage:0)
Behavior when quota exceeded	Notify Only

Budget Control

Budget Editable by Coordinator	False
Reset period	Monthly
	next: 2024.05
Budget (HKD)	20,000 (0.0%) MTD Charge: HKD 0
Notify when budget consumed over	50%
Restrict when budget consumed over	50%
* Budget quota changes may take 60 minutes to apply.	

Kubernetes Quota

Now we set up quota in k8s. we can use subscriptions to control account storage and GPU usage

```
#get all plan  
curl -XGET https://$OASIS_DOMAIN/my-accounts/test/subscription-plans/view \  
-H "Content-Type: application/json" \  
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XGET https://$OASIS_DOMAIN/my-accounts/test/subscription-plans/view \  
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \  
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"  
{  
  "ok":1,"plans":[{"rows":[{"id":"spkk23x","name":"Standard 1GPU - Monthly","type":"monthly","unit":1,"cluster_id":"dev","max_gpu":1,"  
guaranteed_gpu":1,"storage_classes":{"default":100,"standard":0},"max_subscription":20,"price":5600,"currency":"hkd","payment_mechan  
ism":"prepaid","book_in_advance":0,"action":1,"_stats":null},{id":"spk7yes","name":"Trial 1GPU - 2 Weeks","type":"daily","unit":14,  
"cluster_id":"dev","max_gpu":1,"guaranteed_gpu":1,"storage_classes":{"default":50,"standard":0},"max_subscription":10,"price":3000,"  
currency":"hkd","payment_mechanism":"prepaid","book_in_advance":0,"action":1,"_stats":null}],  
"fields":["name","unit","cluster_id","max_gpu","guaranteed_gpu","storage_classes","price","action"],"options":{"itemsPerPage":30,"page":1,"extraColumns":["id","type","book  
_in_advance","payment_mechanism","max_subscription","currency"],"id":"ksp","sortable":["-","action"],"sort":["type","desc"],"filtera  
ble":{"name":"freetext","type":"select","cluster_id":"select","price":"freetext","unit":"freetext","max_gpu":"freetext","guaranteed  
gpu":"freetext","payment_mechanism":"select"},"stats":{"fopts":{"type":["daily","monthly"],"cluster_id":["dev"],"payment_mechanism  
":["prepaid"]},"count":2,"allFields":["id","name","type","unit","cluster_id","max_gpu","guaranteed_gpu","storage_classes","max_subscr  
ption","price","currency","payment_mechanism","book_in_advance","action"],"pagesCount":null,"startAt":1,"endAt":2}}}]}
```

Next, subscribe to a plan.

```
#subscribe a plan to that account
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test/subscription-plans/new \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d
'{"plan":"spk7yes","billing_account_id":"test","activation_start_date":"2022.01.01","auto_renew":"on","timezo
ne":"-480"}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test/subscription-plans/new \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"plan":"spk7yes","billing_account_id":"test","activation_start_date":"2022.01.01","auto_renew":"on"
,"timezone":"-480"}'
{"ok":1}
```

After subscribing, you'll notice that the Kubernetes quota has been updated in the newly created billing account.

Storage Class max storage for each K8s cluster

Storage quota Editable by Coordinator	False
Cluster: dev	2 GB for storage class default 2 GB for storage class standard

GPU Quota for each K8s cluster

GPU Quota Editable by Coordinator	False
Cluster: dev	Guaranteed 8 card Max 8 card

Okay, we have finished configuring the billing account. Let's move on to the consumer account

Storage Class max storage for each K8s cluster

Overview

Storage quota Editable by Coordinator	False
Cluster: dev	2 GB for storage class default 2 GB for storage class standard

Usage: Job

Usage: Storage

GPU Quota for each K8s cluster

Usage: Efficiency

GPU Quota Editable by Coordinator	False
Cluster: dev	Guaranteed 0 card Max 8 card

Settings

Let's begin by assigning a Kubernetes (k8s) storage quota to a specific account. Suppose the k8s cluster has a storage class named 'default'. We plan to allocate 1GB to the test account.

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-storage-quota \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"cluster":"dev","scName":"default","value":"1"}'
```

The result should be like the following:

```
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-storage-quota \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"cluster":"dev","scName":"default","value":"1"}'
{"ok":1}
```

Next, set the Kubernetes (k8s) GPU quota for that account. Assume that you are setting 1 guaranteed GPU and 1 max gpu in the dev cluster for the test account.

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-guaranteed-gpu \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":"1","cluster":"dev"}'
```

```
curl -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-max-gpu \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
-d '{"value":"1","cluster":"dev"}'
```

The result should be like the following:

```

don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-guaranteed-gpu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value": "1", "cluster": "dev"}'
{"ok": 1}%
don@don-F7BSC ~/farmweb % curl -k -XPOST https://$OASIS_DOMAIN/my-accounts/test_child/set-max-gpu \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET" \
don@don-F7BSC ~/farmweb > -d '{"value": "1", "cluster": "dev"}'
{"ok": 1}%

```

finish, that is the web UI display after our work

Accounts ► test_child

Storage Class max storage for each K8s cluster

Overview

Storage quota Editable by Coordinator **False**

Usage: Job

Cluster: dev

1 GB for storage class default
2 GB for storage class standard

Usage: Storage

GPU Quota for each K8s cluster

Usage: Efficiency

GPU Quota Editable by Coordinator **False**

Cluster: dev

Guaranteed 0 card
Max 8 card

Settings

You can also verify this through the API.

```

#verify the consumer account
curl -XGET https://$OASIS_DOMAIN/my-accounts/test_child/settings \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
#verify the setting
curl -XGET https://$OASIS_DOMAIN/my-accounts/test/settings \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"

```

The result should be like the following:

```

don@don-F7BSC ~/farmweb % curl -k -XGET https://$OASIS_DOMAIN/my-accounts/test_child/settings \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
{"ok":1,"acct":{"id":"test_child","parent":"test","owner_email":"test2@test.com","type":"consumer","status":"active","config":{"gid":10000,"fairshare":100,"created_at":"2024-05-09T02:05:10.796Z","updated_at":"2024-05-09T02:26:22.319Z","user_quota":10,"guaranteed_gpu":{"dev":1},"max_gpu":{"dev":1},"storage_class_max_gb":{"dev":{"default":1,"standard":0}},"charge_ratio":null,"budget_exceed_restricted":false,"budget_config":{"budget":-1},"storage_charge_ratio":null,"eff_settings":{"start_cal":10,"notify_owner":true,"cpu_min_threshold":50,"mem_min_threshold":50,"notify_coordinator":false},"quota":{"usage":{"last_reset_time":"2024-05-09T02:05:10.796Z"},"reset_period":"none","on_quota_exceeded":"notify_only","parent_account_coordinator":"test","is_coordinator":null,"membership_config":null},"status":"active","quotaDetail":{"id":"test_child","status":"active","guaranteed_gpu":{"dev":1},"max_gpu":{"dev":1},"storage_class_max_gb":{"dev":{"default":1,"standard":0}},"reset_period":"none","_next_reset_date":null,"builtin_csu":null,"shared_csu":null,"shared_gsu":null,"test_csu":null,"_quota":{"users":"0","_user_quota":10,"_is_coordinator":1},"quotaMapping":{"builtin_csu":"CPU Builtin","builtin_gsu":"GPU Builtin","shared_csu":"CPU Shared","shared_gsu":"GPU Shared","test_csu":"CPU Test","test_gsu":"GPU Test"},"resetPeriodOptions":["now","none","daily","weekly","monthly","quarterly","yearly"],"onQuotaExceededOption":["no_action","notify_only","auto_kill_jobs"],"quotaEditable":true,"budgetInfo":{"currentUsedHkd":0,"budgetEditable":true,"exceeded_notify_ratio":1,"exceeded_restrict_ratio":1,"budget":-1},"k8sClusters":[{"id":"dev","storage_classes":[{"id":"default","maxGb":500,"isDefault":true},{id":"standard","maxGb":500,"isDefault":false}]}],"guaranteedGpuQuota":{"dev":1},"maxGpuQuota":{"dev":1},"scMaxGb":{"dev":{"default":1,"standard":0}}}}
don@don-F7BSC ~/farmweb % curl -k -XGET https://$OASIS_DOMAIN/my-accounts/test/settings \
don@don-F7BSC ~/farmweb > -H "Content-Type: application/json" \
don@don-F7BSC ~/farmweb > -H "Authorization: Bearer $OASIS_KEY:$OASIS_SECRET"
{"ok":1,"acct":{"id":"test","parent":"oneasia","owner_email":"test@test.com","type":"billing","status":"active","config":{"gid":4001,"owner_name":"test","quota_editable_by_coordinator":false,"budget_editable_by_coordinator":false,"fairshare":100,"created_at":"2024-05-09T02:05:07.780Z","updated_at":"2024-05-09T02:20:56.534Z","user_quota":10,"guaranteed_gpu":{"dev":1},"max_gpu":{"dev":1},"storage_class_max_gb":{"dev":{"default":50,"standard":0}},"charge_ratio":null,"budget_exceed_restricted":false,"budget_config":{"budget":20000,"exceeded_notify_ratio":0.5,"exceeded_restrict_ratio":0.5},"storage_charge_ratio":null,"eff_settings":{"start_cal":10,"notify_owner":true,"cpu_min_threshold":50,"mem_min_threshold":50,"notify_coordinator":false},"quota":{"shared_csu":50,"shared_gsu":50,"builtin_csu":50,"builtin_gsu":50},"usage":{"last_reset_time":"2024-05-09T02:05:07.780Z"},"reset_period":"none","on_quota_exceeded":"notify_only","parent_account_coordinator":"oneasia","is_coordinator":true,"membership_config":{"status":"active","quotaDetail":{"id":"test","status":"active","guaranteed_gpu":{"dev":1},"max_gpu":{"dev":1},"storage_class_max_gb":{"dev":{"default":50,"standard":0}},"reset_period":"none","_next_reset_date":null,"builtin_csu":null,"shared_csu":null,"shared_gsu":null,"test_csu":null,"test_gsu":null,"_quota":{"shared_csu":50,"shared_gsu":50,"builtin_csu":50,"builtin_gsu":50},"users":"2","_user_quota":10,"_is_coordinator":1},"quotaMapping":{"builtin_csu":"CPU Builtin","builtin_gsu":"GPU Builtin","shared_csu":"CPU Shared","shared_gsu":"GPU Shared","test_csu":"CPU Test","test_gsu":"GPU Test"},"resetPeriodOptions":["now","none","daily","weekly","monthly","quarterly","yearly"],"onQuotaExceededOption":["no_action","notify_only","auto_kill_jobs"],"quotaEditable":true,"budgetInfo":{"currentUsedHkd":0,"budgetEditable":true,"exceeded_notify_ratio":0.5,"exceeded_restrict_ratio":0.5,"budget":20000},"k8sClusters":[{"id":"dev","storage_classes":[{"id":"default","maxGb":500,"isDefault":true},{id":"standard","maxGb":500,"isDefault":false}]}],"guaranteedGpuQuota":{"dev":1},"maxGpuQuota":{"dev":1},"scMaxGb":{"dev":{"default":50,"standard":0}}}}

```

Revision #5

Created 7 May 2024 08:13:49 by Don Chu

Updated 10 May 2024 01:33:58 by Don Chu