

Prerequisites

Remote machine 1: VM with Ubuntu, CapRover and fixed IP*

Remote machine 2: VM with Ubuntu, CapRover and fixed IP*

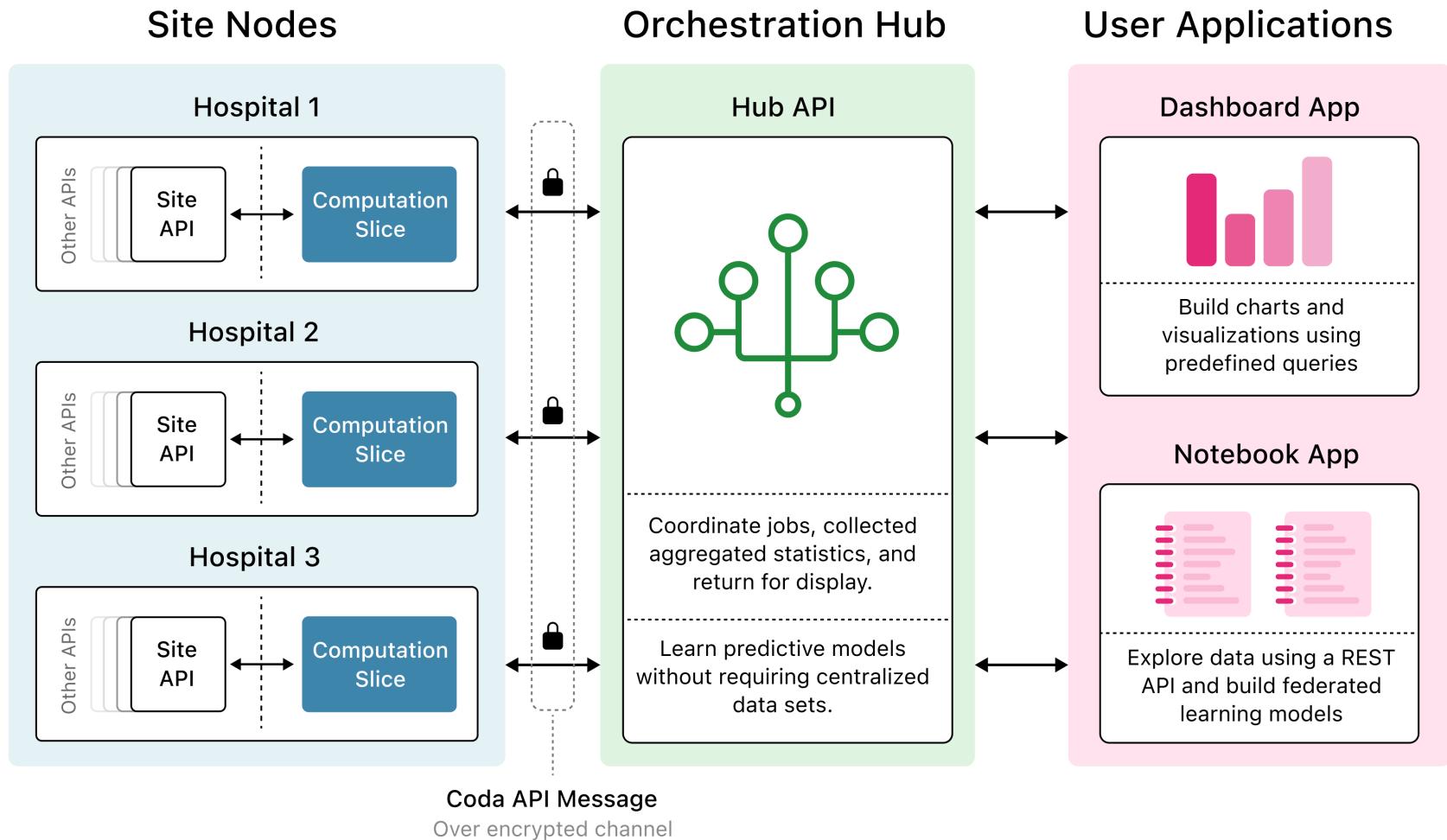
(Your) local machine: [GitHub](#) and [NodeJS >= 18](#) installed

A domain name: with access to DNS configuration settings

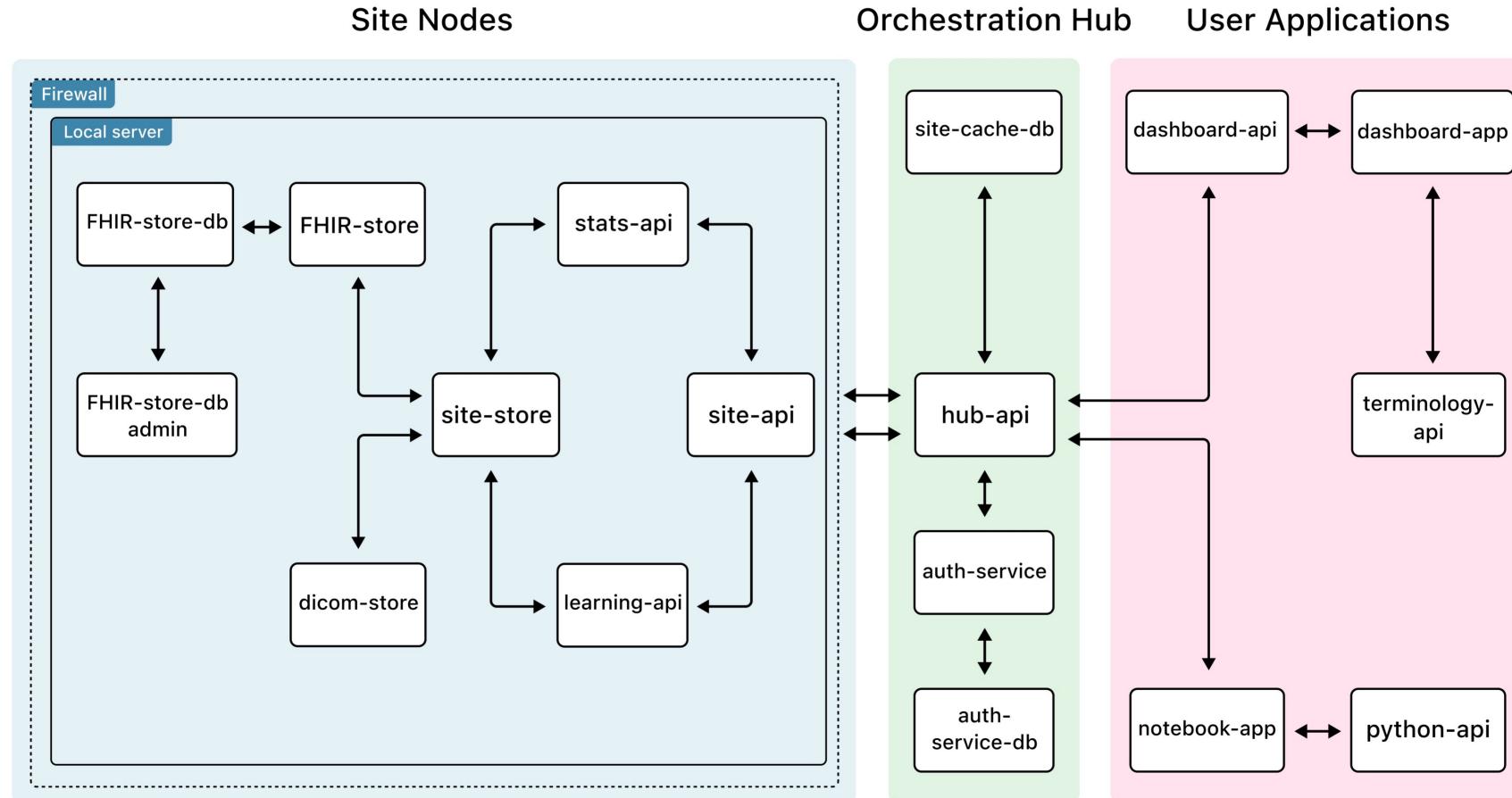
*For example, using one-click deploy on DigitalOcean:

<https://caprover.com/docs/get-started.html>

Conceptual overview



Technical overview

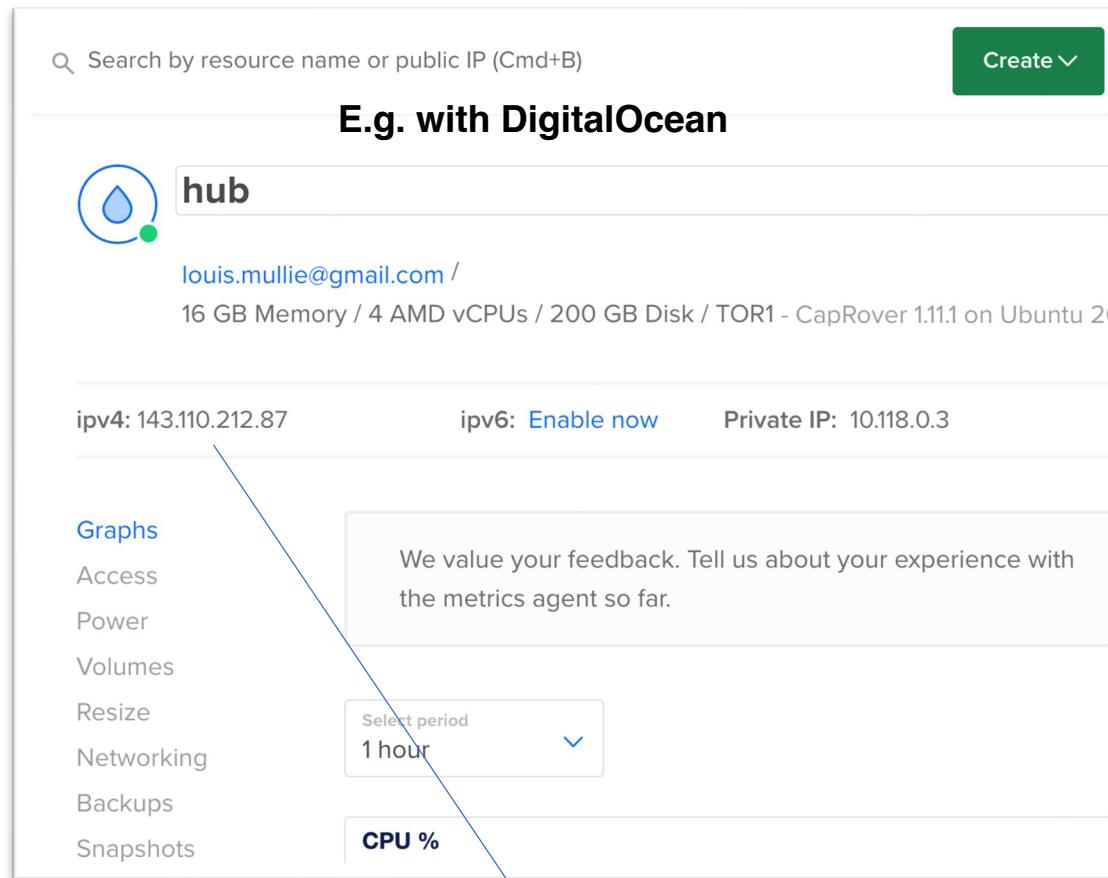


Each white box corresponds to a GitHub repository:
<https://github.com/orgs/coda-platform/repositories>

1. Configure CapRover

The process will be demonstrated for the **hub**
and is identical for the all nodes.

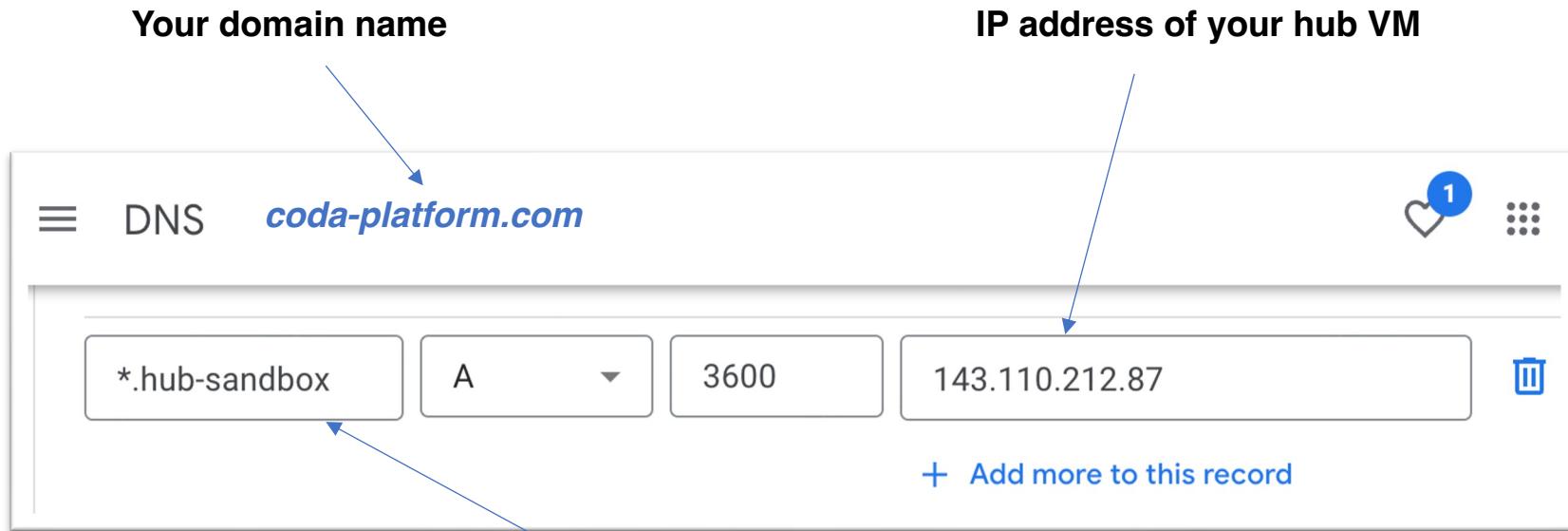
1.1 Obtain the IP of the machine you will be using for the hub



**Note the IP address of your hub server.
In this example, it is:**

143.110.212.87

1.2 Configure DNS on your domain name

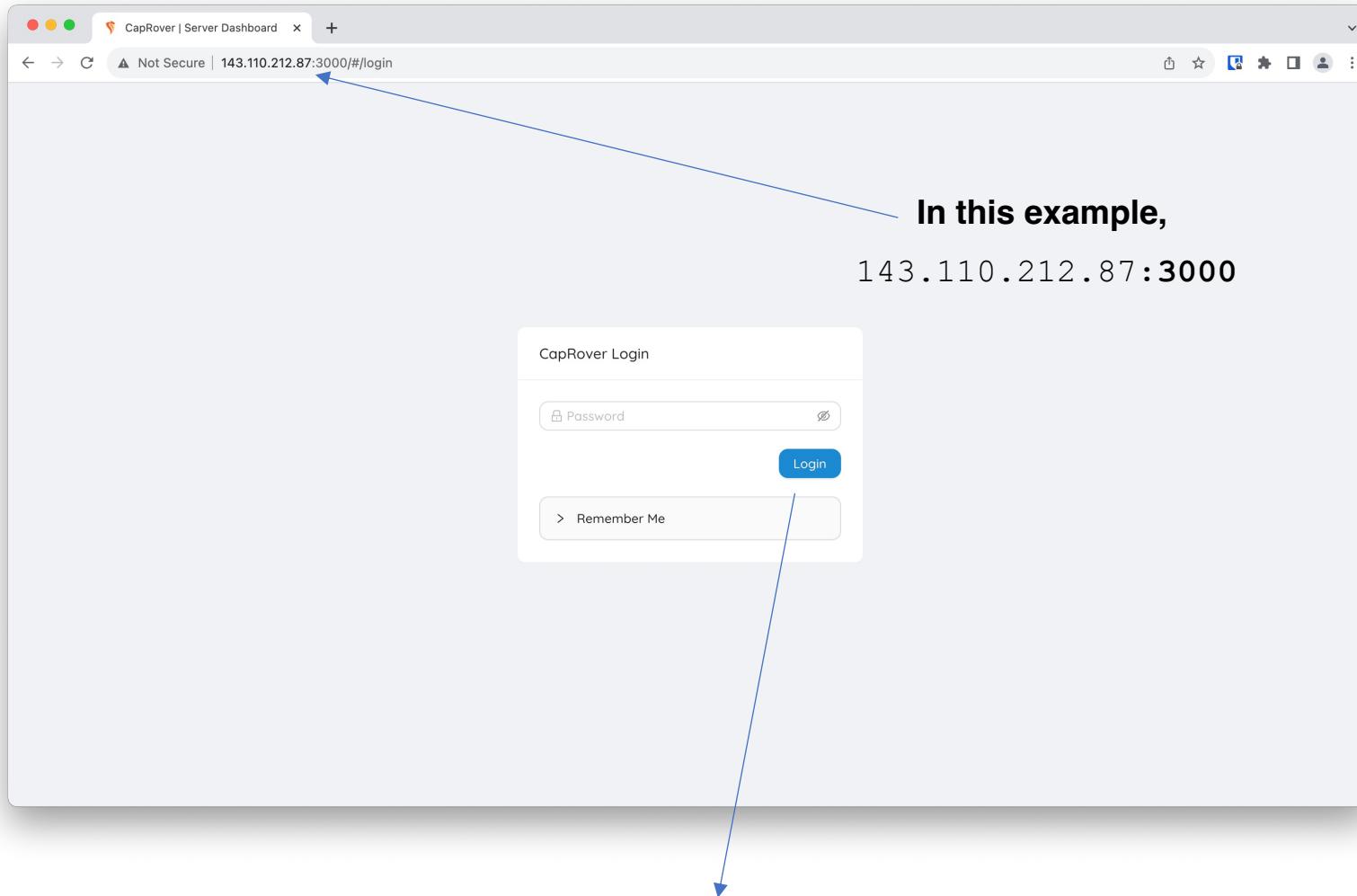


Subdomain of your choice for hub

In this example, hub will live at:

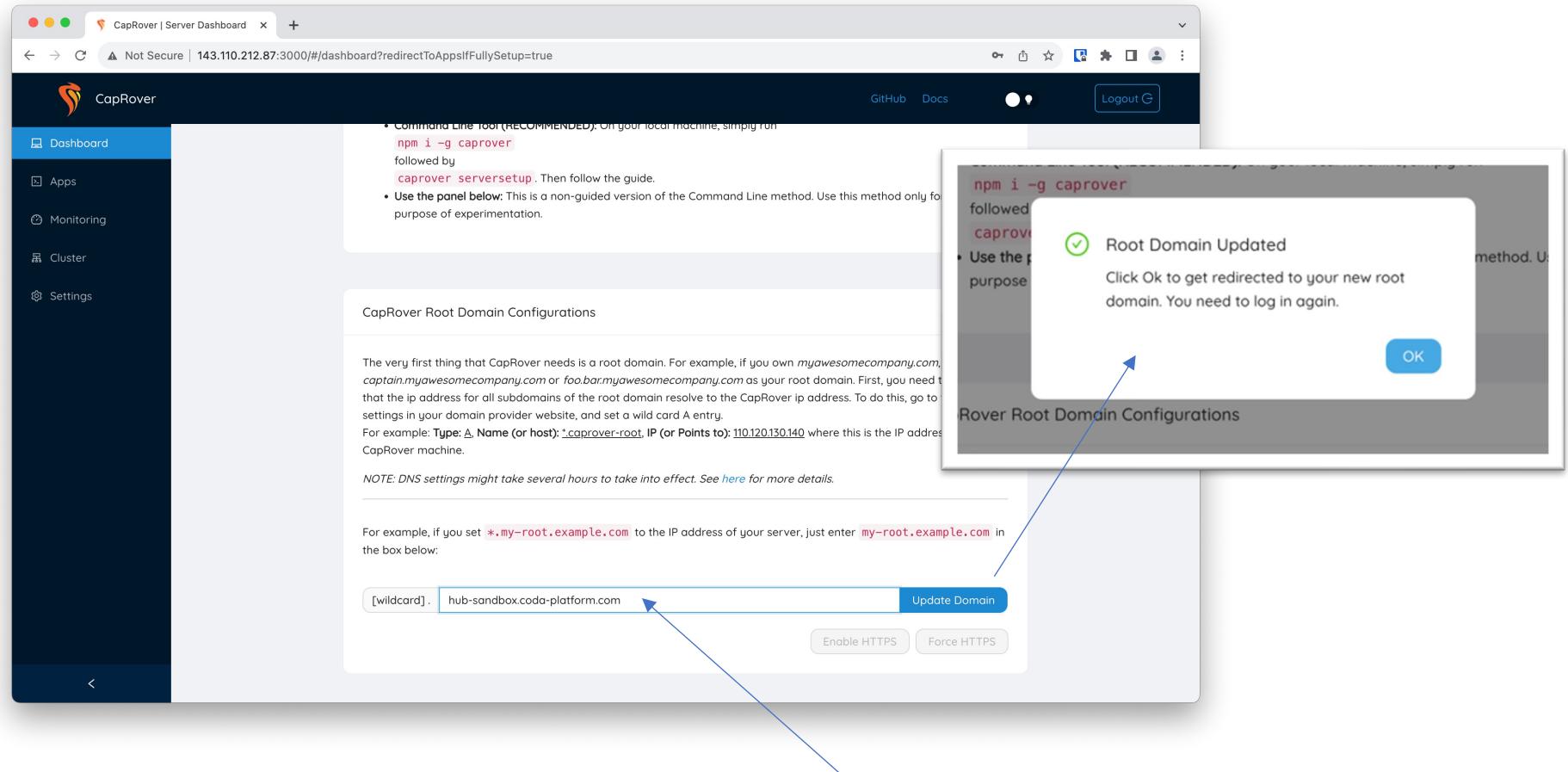
* .**hub-sandbox**.coda-platform.com

1.3 Navigate to http://<server IP>:3000



Login using default password: captain42

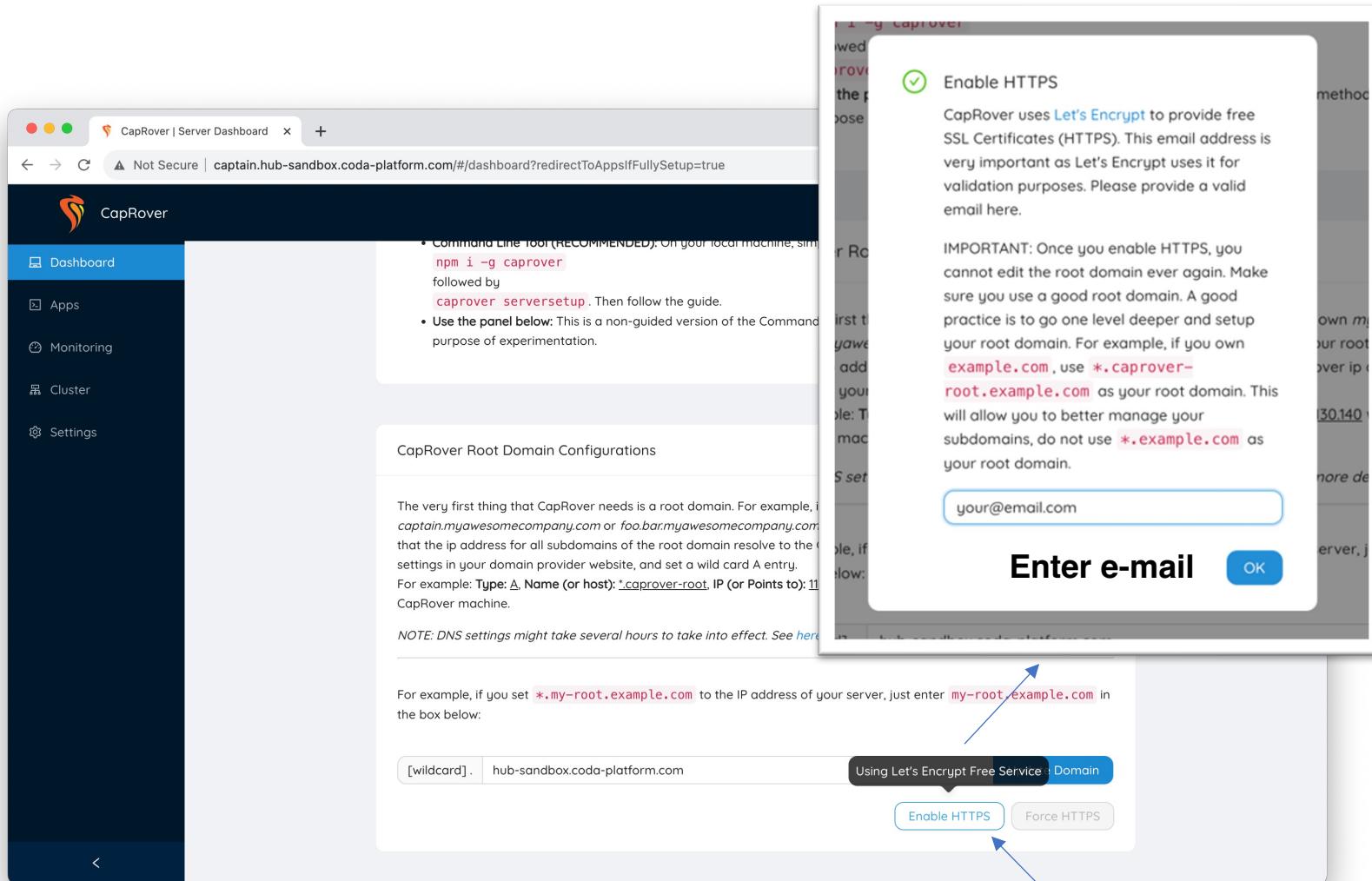
1.4 Configure CapRover root domain



Enter root domain that will be used for “hub” server
In this example, we previously configured DNS for:

hub-sandbox.coda-platform.com

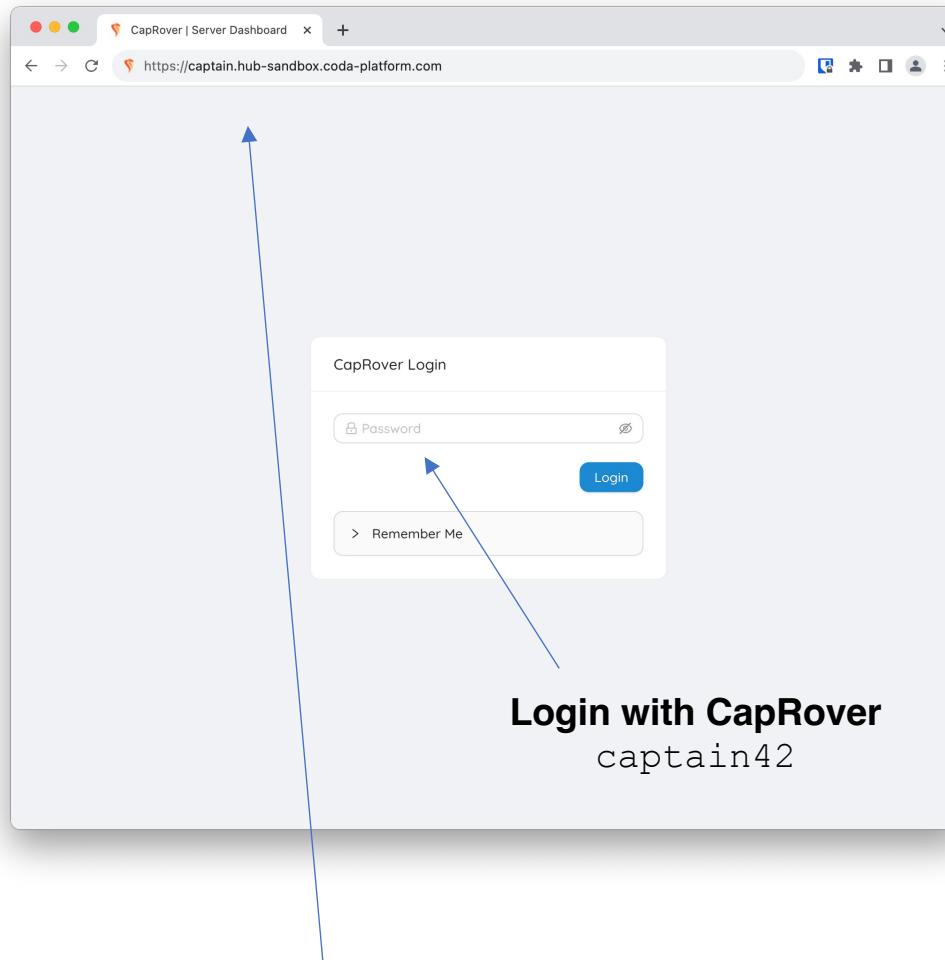
1.5 Enable HTTPS



Enable HTTPS

1.6 Test root domain with HTTPS

<https://captain.your-hub-subdomain.your-domain.com>

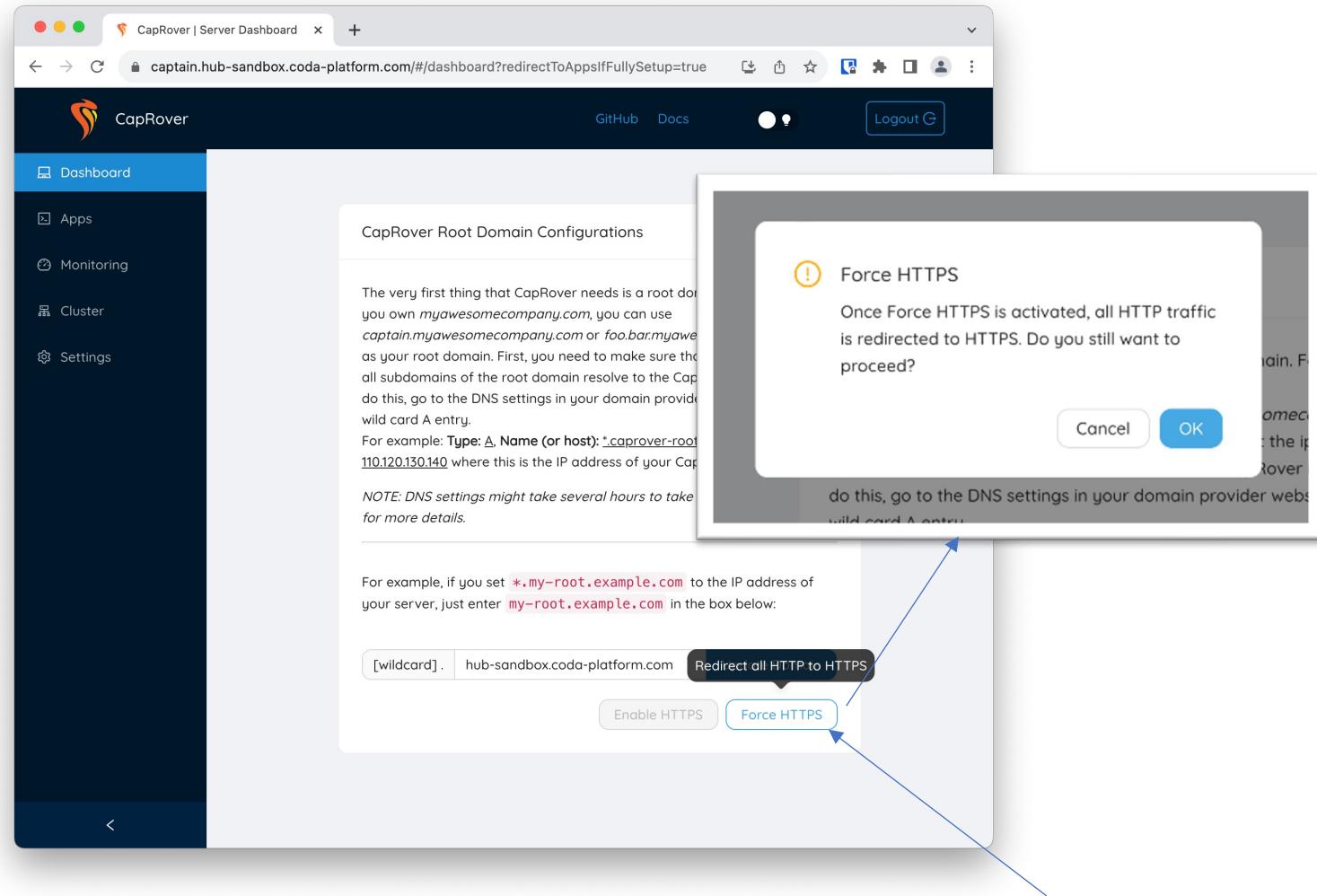


Navigate to the address as described above

In this example, it corresponds to:

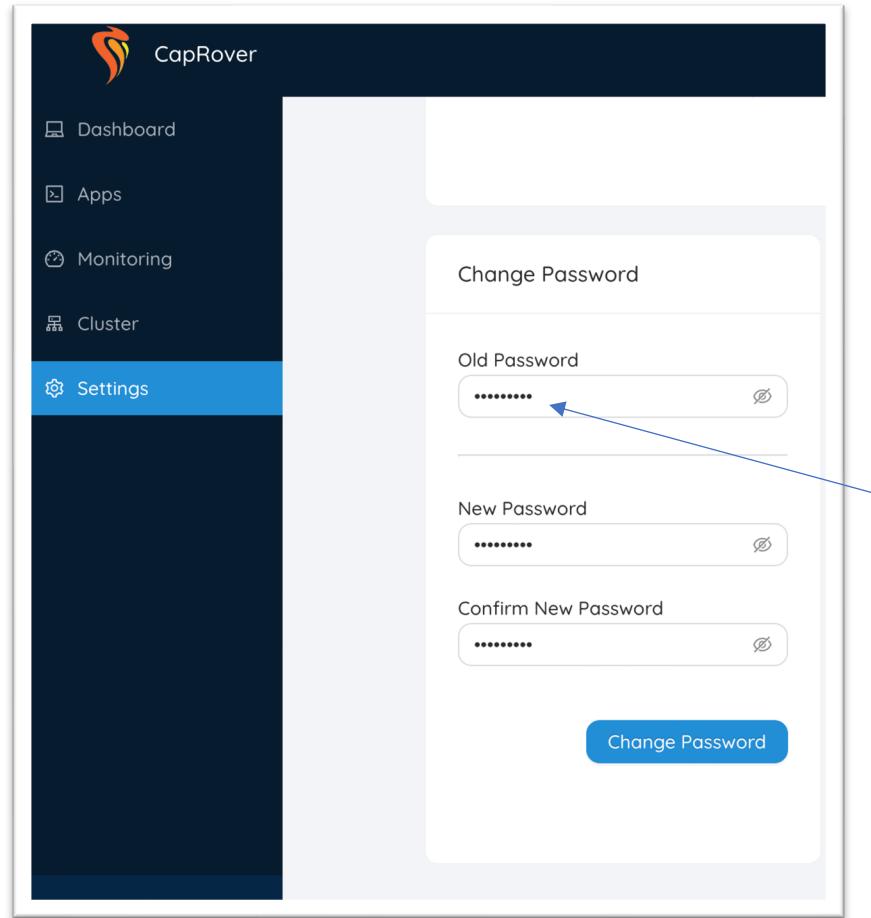
<https://captain.hub-sandbox.coda-platform.com>

1.7 Force HTTPS on the root domain



Force HTTPS

1.8 Change the default password



Default password
captain42

2. Configure CapRover on a site machine

Repeat steps 1-9 using a different server (IP) and subdomain.

In this example, we'll be using the following for site #1:

Site 1 IP address: 165.227.38.118

Site 1 root domain: sitel-sandbox.coda-platform.com

Site 1 DNS A record: *.sitel-sandbox.coda-platform.com A 165.227.38.118

3. Configure additional machines for multi-site

Repeat previous steps as for site 1, modifying configuration accordingly.

Make sure you update the following configuration variables when deploying additional sites.

```
CAPROVER_NAME, CAPROVER_URL,  
CODA_SITE_API_HOSPITAL_CODE,  
CODA_SITE_API_AUTH_USERNAME,  
CODA_SITE_API_AUTH_PASSWORD
```

Overview at this stage

Your configuration should look similar to this
(substituting your IPs, domain name, and subdomains).

VM	IP in A record	Root domain	CapRover control panel
0	143.110.212.87	hub-sandbox.coda-platform.com	captain.hub-sandbox.coda-platform.com
1	165.227.38.118	sitel-sandbox.coda-platform.com	captain.sitel-sandbox.coda-platform.com
...
4		site4-sandbox.coda-platform.com	captain.site4-sandbox.coda-platform.com

4. Deploy the hub

Clone the hub deployment automation scripts:

<https://github.com/coda-platform/hub-deployer>

Then, navigate to the directory where it was cloned in your terminal.

Once inside this directory, follow the steps on the next slide.

Configure environment

in `.env` file →

```
CAPROVER_URL=https://captain.hub-sandbox.coda-platform.com  
CAPROVER_PASSWORD=captain42  
CAPROVER_NAME=hub-sandbox  
CAPROVER_ROOT=coda-platform.com  
  
CODA_AUTH_SERVICE_URL=https://auth-service.hub-sandbox.coda-platform.com  
CODA_HUB_API_URL=https://hub-api.hub-sandbox.coda-platform.com  
CODA_DASHBOARD_API_URL=https://dashboard-api.hub-sandbox.coda-platform.com  
CODA_DASHBOARD_APP_URL=https://dashboard-app.hub-sandbox.coda-platform.com
```

Run `npm install` →

```
(base) louismullie@Louis-Antoines-MacBook-Pro-2 hub-deployer % npm install  
up to date, audited 10 packages in 216ms  
1 package is looking for funding  
  run `npm fund` for details  
found 0 vulnerabilities  
(base) louismullie@Louis-Antoines-MacBook-Pro-2 hub-deployer % █
```

Once done, run:
`npm run deploy`

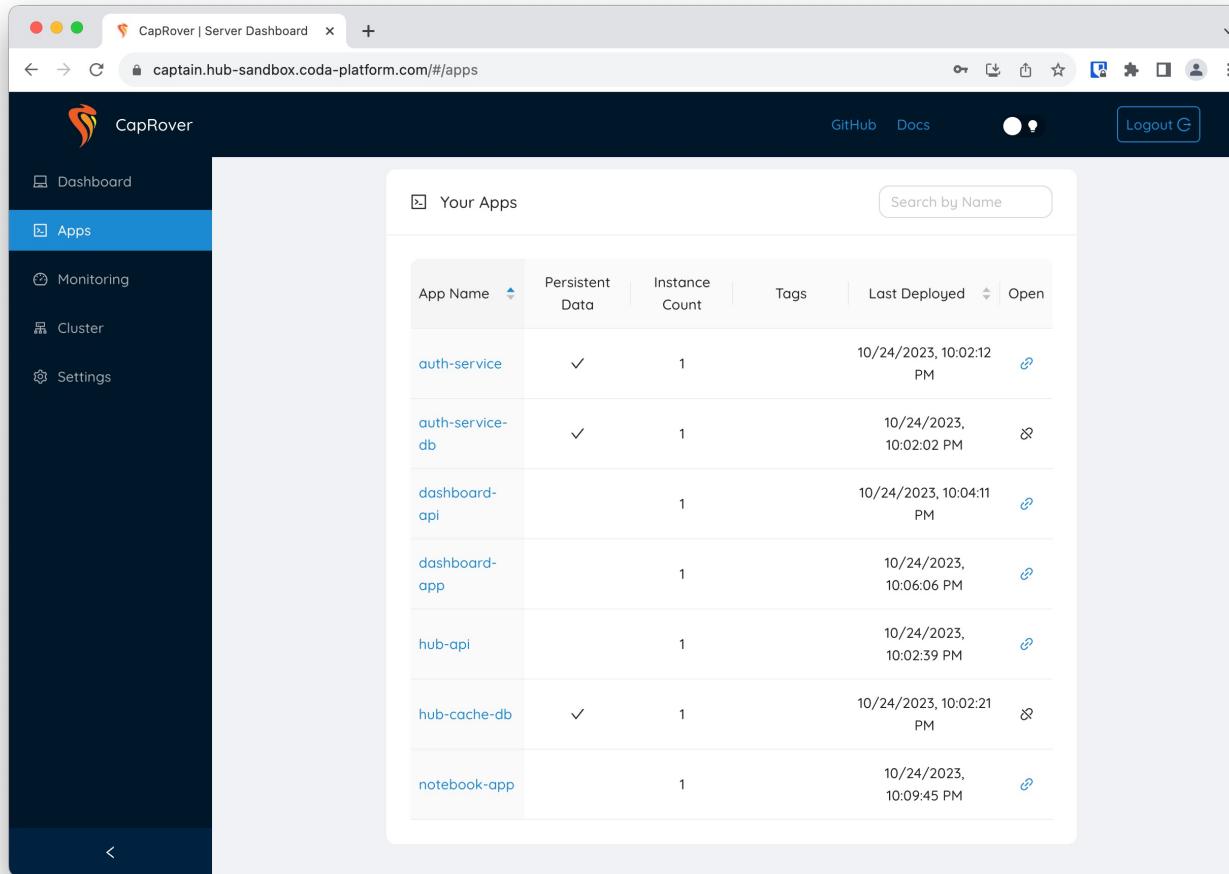
Run `caprover login` →

Using password you created for site machine during CapRover setup.

```
(base) louismullie@Louis-Antoines-MacBook-Pro-2 hub-deployer % caprover login  
  
Login to a CapRover machine...  
  
? CapRover machine URL address, it is "[http[s]://][captain.]your-captain-root.domain": https://captain.hub-sandbox.coda-platform.com  
? CapRover machine password: [hidden]  
? CapRover machine name, with whom the login credentials are stored locally: hub-sandbox  
  
Logged in successfully.  
Authorization token is now saved as hub-sandbox at https://captain.hub-sandbox.coda-platform.com.
```

Open the hub CapRover control panel

<https://captain.your-hub-subdomain.your-domain.com>



Your dashboard should show the exact same application names and configurations.

5. Deploy a site

Clone the site deployment automation scripts:

<https://github.com/coda-platform/site-deployer>

Then, navigate to the directory where it was cloned in your terminal.

Once inside this directory, follow the steps on the next slide.

**Configure environment
in .env file →**

```
CAPROVER_URL=https://captain.site1-sandbox.coda-platform.com
CAPROVER_PASSWORD=captain42
CAPROVER_NAME=site1-sandbox

CODA_AUTH_SERVICE_URL=https://auth-service.hub-sandbox.coda-platform.com
CODA_HUB_API_URL=https://hub-api.hub-sandbox.coda-platform.com

CODA_STATS_API_URL=http://srv-captain--stats-api:8082
CODA_LEARNING_API_URL=http://srv-captain--learning-api:8084
CODA_FHIR_STORE_URL=http://srv-captain--fhir-store:8888
CODA_DICOM_STORE_URL=http://srv-captain--dicom-store:8042
```

Run npm install →

```
(base) louismullie@Louis-Antoines-MacBook-Pro-2 site-deployer % npm install
up to date, audited 10 packages in 226ms

1 package is looking for funding
  run `npm fund` for details

found 0 vulnerabilities
(base) louismullie@Louis-Antoines-MacBook-Pro-2 site-deployer % █
```

Run caprover login →
Using password you
created for hub
machine during
CapRover setup.

```
(base) louismullie@Louis-Antoines-MacBook-Pro-2 site-deployer % caprover login

Login to a CapRover machine...
? CapRover machine URL address, it is "[http[s]://][captain.]your-captain-root.domain": https://captain.site1-sandbox.coda-platform.com
? CapRover machine password: [hidden]
? CapRover machine name, with whom the login credentials are stored locally: site1-sandbox

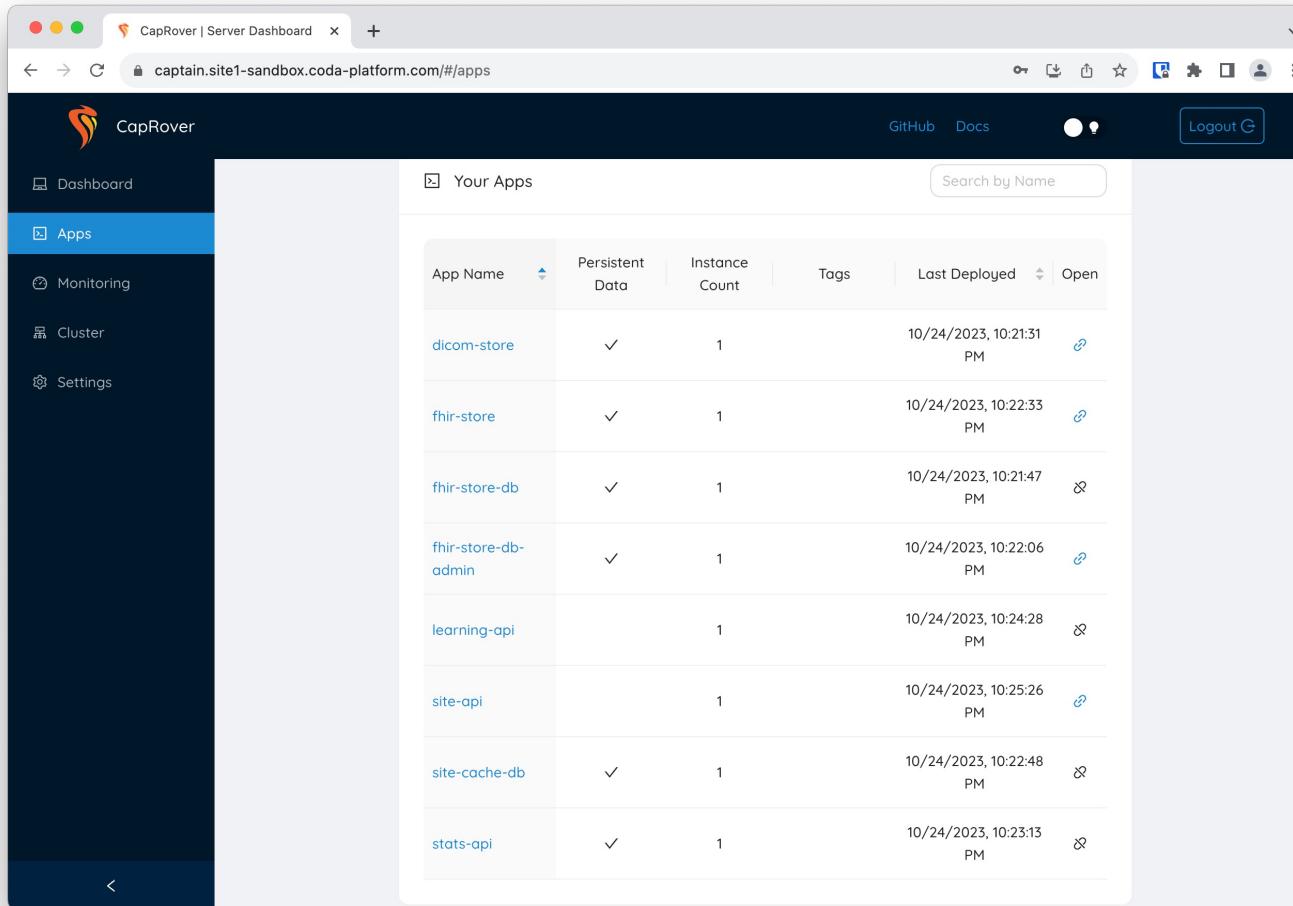
Logged in successfully.
Authorization token is now saved as site1-sandbox at https://captain.site1-sandbox.coda-platform.com.

(base) louismullie@Louis-Antoines-MacBook-Pro-2 site-deployer % █
```

**Once done, run:
npm run deploy**

Open the site CapRover control panel

`https://captain.your-site-subdomain.your-domain.com`



The screenshot shows the CapRover Server Dashboard interface. The left sidebar has a dark theme with blue highlights for the 'Apps' section, which is currently selected. The main content area is titled 'Your Apps' and displays a table with the following data:

App Name	Persistent Data	Instance Count	Tags	Last Deployed	Open
dicom-store	✓	1		10/24/2023, 10:21:31 PM	🔗
fhir-store	✓	1		10/24/2023, 10:22:33 PM	🔗
fhir-store-db	✓	1		10/24/2023, 10:21:47 PM	🔗
fhir-store-db-admin	✓	1		10/24/2023, 10:22:06 PM	🔗
learning-api		1		10/24/2023, 10:24:28 PM	🔗
site-api		1		10/24/2023, 10:25:26 PM	🔗
site-cache-db	✓	1		10/24/2023, 10:22:48 PM	🔗
stats-api	✓	1		10/24/2023, 10:23:13 PM	🔗

Your dashboard should show the exact same application names and configurations.

6. Configure Keycloak

Open the **hub** CapRover control panel

<https://captain.your-hub-subdomain.your-domain.com>

Open auth-service app to get link to Keycloak

The screenshot shows the CapRover dashboard with the auth-service application selected. Under the "HTTP Settings" tab, it displays the internal URL as `srv-captain--auth-service` and the public URL as `https://auth-service.hub-sandbox.coda-platform.com`. There are options for enabling HTTPS, connecting new domains, and editing Nginx configurations. The container HTTP port is set to 8080.

The screenshot shows the "Welcome to Keycloak" page. It features links to the Administration Console, Documentation, and Keycloak Project. At the bottom, there are links for Mailing List and Report an issue. A blue arrow points from the public URL in the CapRover screenshot to the "Administration Console" link in the Keycloak screenshot.

Admin username and password defined in app environment

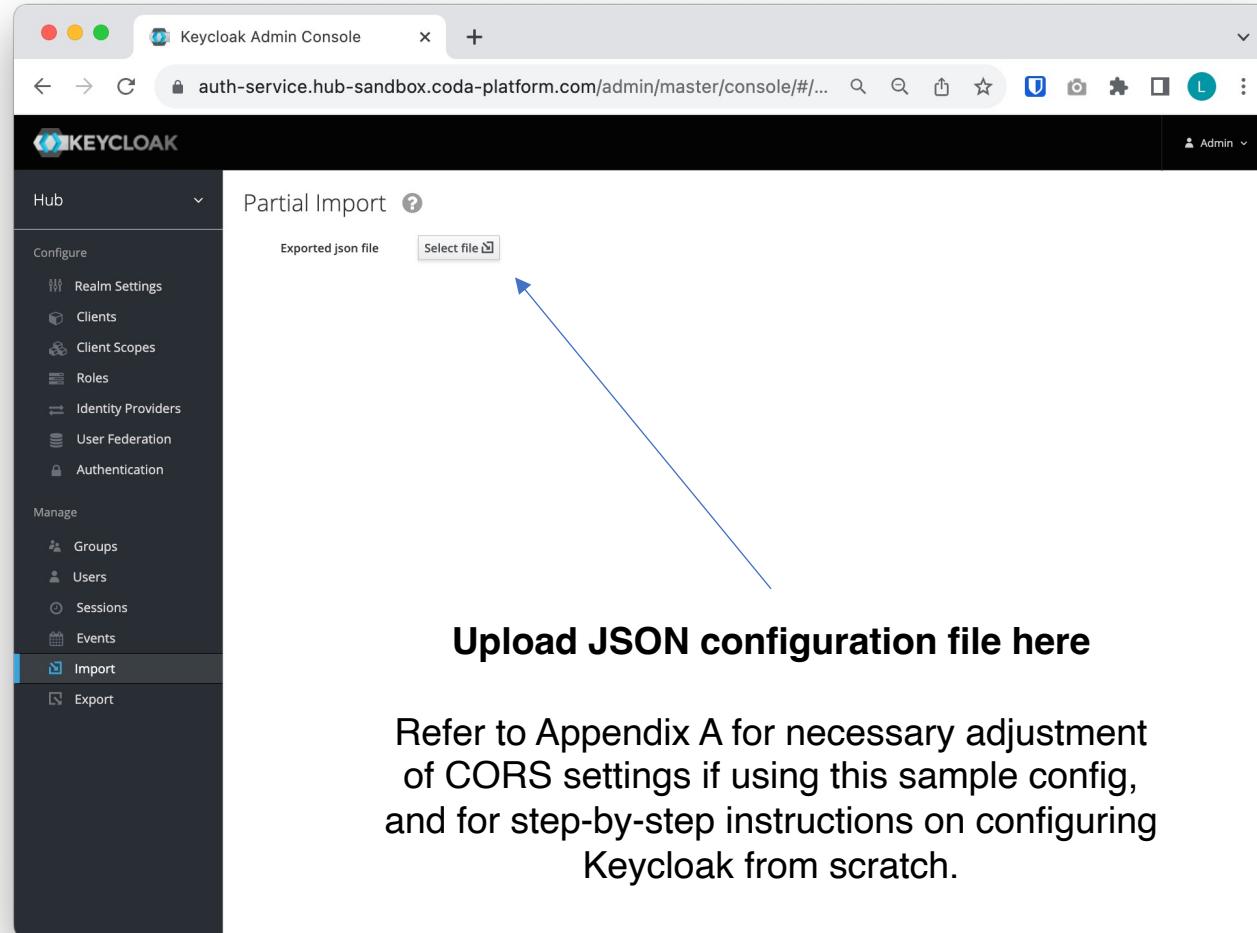
CODA_AUTH_SERVICE_ADMIN_USERNAME
CODA_AUTH_SERVICE_ADMIN_PASSWORD

The screenshot shows the auth-service application settings under the "App Configs" tab. It lists environmental variables: CODA_AUTH_SERVICE_PORT (8080), CODA_AUTH_SERVICE_ADMIN_USERNAME (admin), and CODA_AUTH_SERVICE_ADMIN_PASSWORD (unsafeunsafe).

The screenshot shows the "Sign in to your account" page of the Keycloak administration console. It has fields for "Username or email" and "Password", and a "Sign In" button. Below the form, the text "Default: admin / unsafeunsafe" is displayed. A blue arrow points from the CODA_AUTH_SERVICE_ADMIN_USERNAME variable in the app config to the "Username or email" field, and another arrow points from the CODA_AUTH_SERVICE_ADMIN_PASSWORD variable to the "Password" field.

Configure Keycloak using a default / example configuration

<https://github.com/coda-platform/auth-service/blob/main/config.example.json>



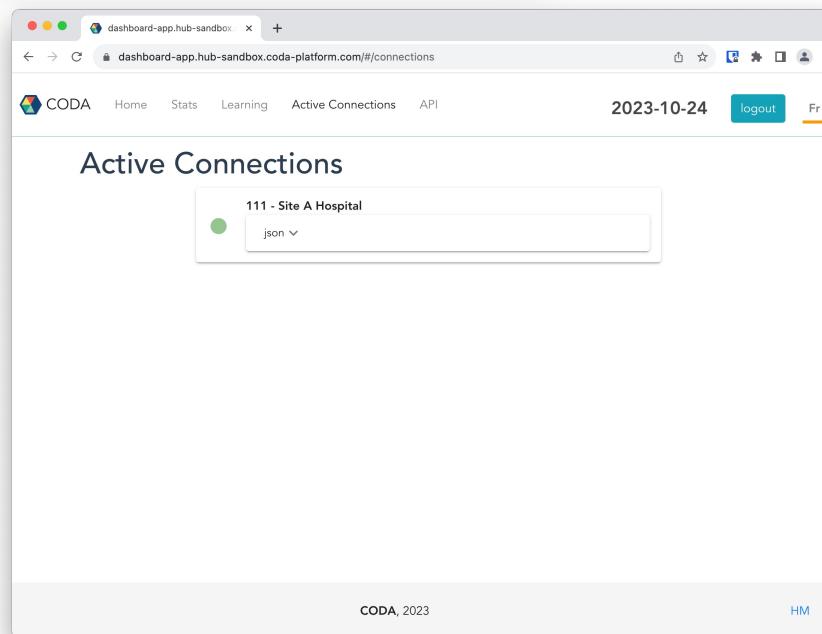
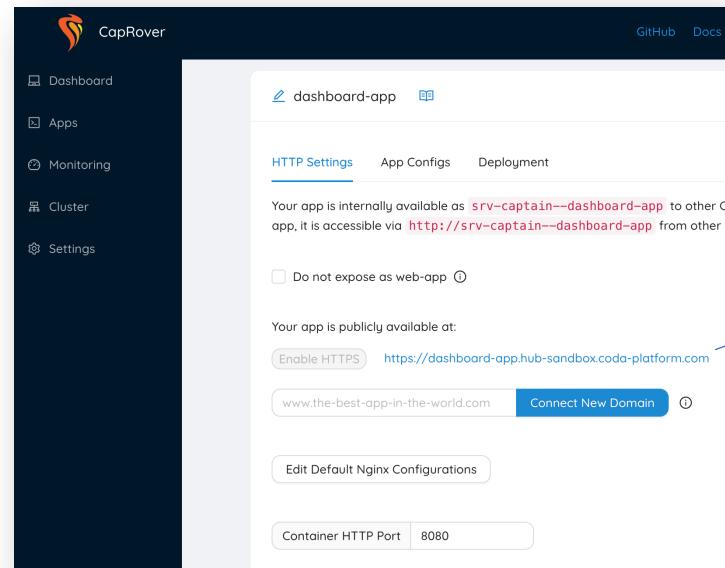
Note: the “sandbox” configuration is insecure, and for testing purposes only.

7. Test dashboard login

Open the **hub** CapRover control panel

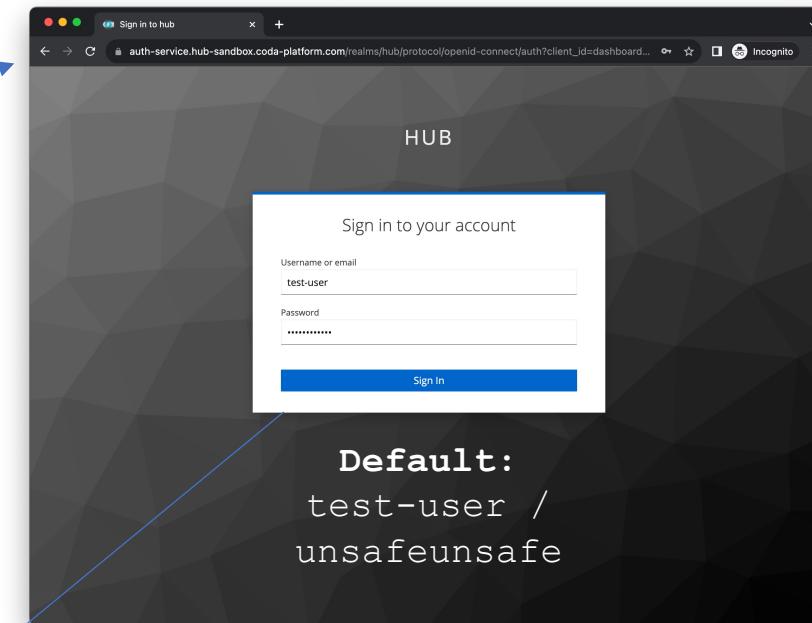
<https://captain.your-hub-subdomain.your-domain.com>

Open dashboard-app app to get link to dashboard



Login using test-user credentials, i.e.

CODA_DASHBOARD_APP_AUTH_CLIENT_ID
CODA_DASHBOARD_API_AUTH_CLIENT_SECRET



Confirm that able to successfully login to the dashboard. If Keycloak is properly configured, you should see that the site you created has already connected to the hub.

8. Load dummy FHIR data

Open the **site** CapRover control panel

<https://captain.your-site-subdomain.your-domain.com>

Open fhir-store app to get link to AidBox

The screenshot shows the CapRover dashboard with the fhir-store app selected. Under the 'HTTP Settings' tab, it displays the internal URL as 'srv-captain--fhir-store' and the publicly available URL as 'https://fhir-store.site1-sandbox.coda-platform.com'. A blue arrow points from this URL to the login page of the AidBox interface.

Login using FHIR store credentials, i.e.

CODA_FHIR_STORE_DB_USERNAME
CODA_FHIR_STORE_DB_PASSWORD

The screenshot shows the AidBox login page with the URL 'fhir-store.site1-sandbox.coda-platform.com/auth/login'. It features a logo, 'Log in', and 'Sign up' buttons. The 'Username or Email' field contains 'admin' and the 'Password' field contains 'unsafeunsafe'. A blue arrow points from the AidBox login page back to the REST console.

Default:
admin /
unsafeunsafe

The screenshot shows the AidBox REST Console with the URL 'fhir-store.site1-sandbox.coda-platform.com/ui/console#/rest'. It includes a sidebar with various menu items like Notebooks, Profiles, Resources, Operations, REST Console, DB Console, DB Tables, DB Queries, Attrs Stats, Apps, Auth Clients, Users, Access Control, GraphQL, and Auth Sandbox. The main area shows a snippet of code for loading Synthea data via a POST request to '/fhir/\$load'.

**Confirm that able to login to Aidbox.
Go to REST console in menu, load Synthea data:**

```
POST /fhir/$load
Accept: text/yaml
Content-Type: text/yaml

source: 'https://storage.googleapis.com/aidbox-public/synthea/100/all.ndjson.gz'
```

Expected response when Synthea data imported successfully:

REST Console snippet name **Save to Snippets** **Send**

```
POST /fhir/$load
Accept: text/yaml
Content-Type: text/yaml

source: 'https://storage.googleapis.com/aidbox-public/synthea/100/all.ndjson.gz'
```

YAML Snippet **JSON Snippet** Press **Ctrl + Enter** to run query

Status: 200 **Pretty** **Raw**

Headers

```
CarePlan: 356
Observation: 20382
MedicationAdministration: 150
Goal: 301
Patient: 124
DiagnosticReport: 1430
Practitioner: 181
ExplanationOfBenefit: 3460
Immunization: 1636
Claim: 4488
MedicationRequest: 1028
Encounter: 3460
Condition: 871
Procedure: 2854
```

9. Test federated analytics

Open the **dashboard** application URL

`https://dashboard-app.your-hub-subdomain.your-domain.com`

Login using **test-user / unsafeunsafe** if you haven't changed the defaults.

Querying age by site after creating 3 site nodes and loading them with variations of the same FHIR dataset.

CODA Home Stats Learning Active Connections API 04/09/2021 logout Fr

Select hospitals to include:
Site A Hospital ✕ Site B Hospital ✕ Site C Hospital ✕

Select measures to retrieve:
Continuous variables (e.g. age)
count ✕ mean ✕ stdev ✕ ci95 ✕
Discrete (e.g. gender)
count ✕ mode ✕

Resources
Patient ✕ Add (+)
Filters
AND OR Add filter Add group
Fields
age

Breakdown
Resource type Patient
Resource attribute age
Start (lower limit) 0 End (upper limit) 100 Stride (step size) 20

Results
Table 1. Distribution of age by site

0	20	40	60	80	Site	Total
57	27	246	572	84	Site C	988
36	11	243	623	87	Site B	1000
61	32	280	653	96	Site A	1124
154	70	769	1848	267	All	3112

Figure 1. Distribution of age by site

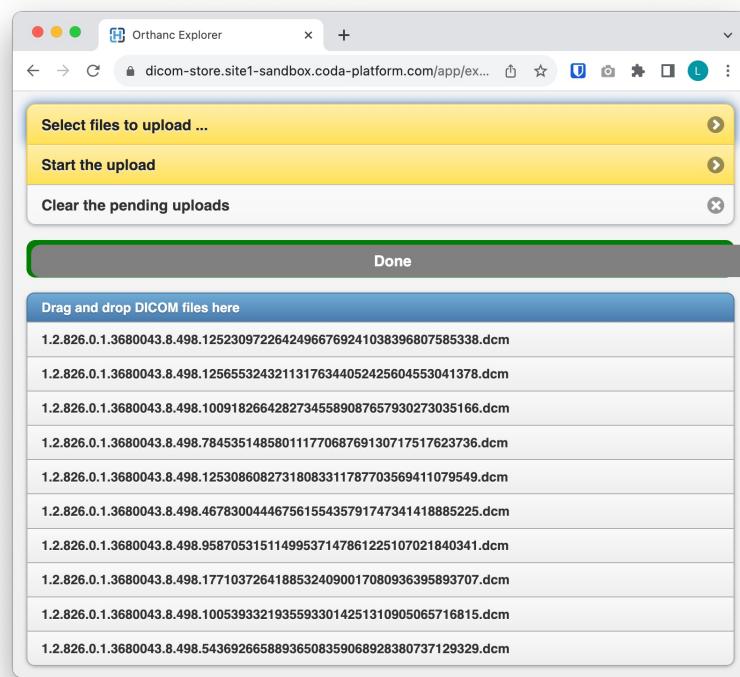
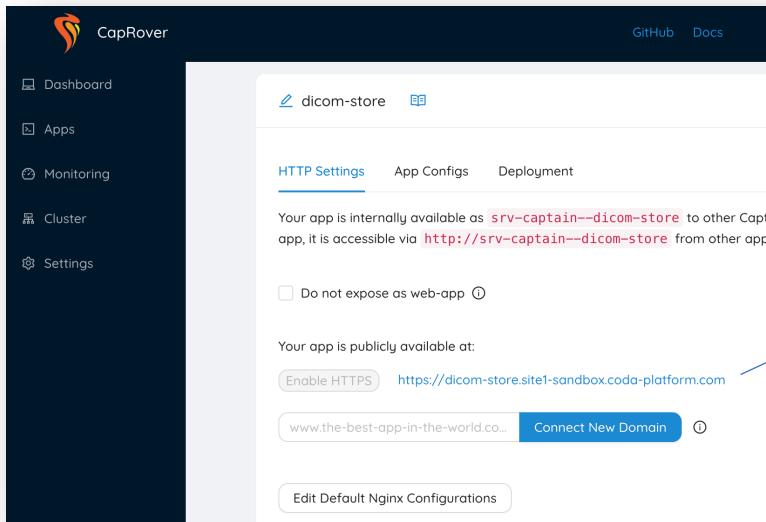
Age Group	Site A	Site B	Site C	Total
0	57	27	11	988
20	36	11	243	1000
40	61	32	280	1124
60	154	70	769	3112
80	267	0	0	0

10. Load dummy DICOM data

Open the **site** CapRover control panel

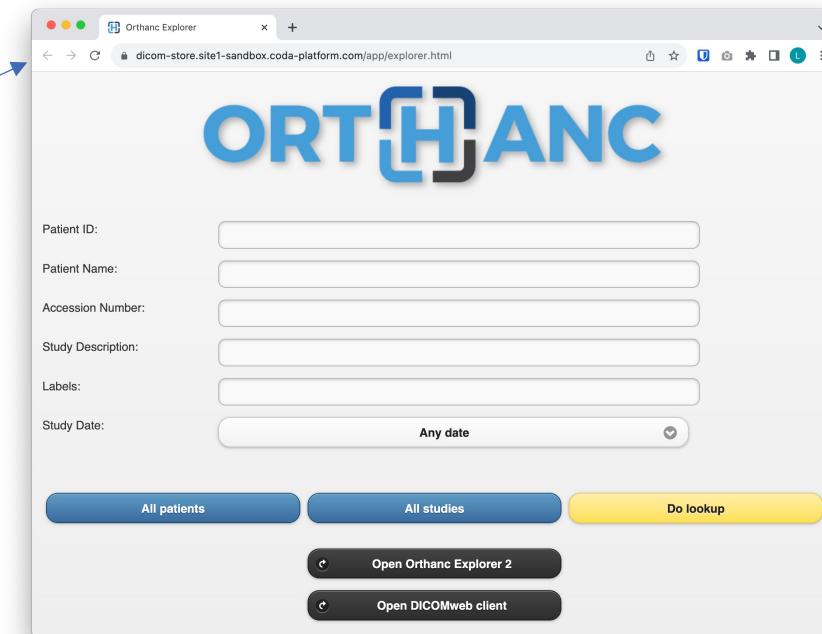
<https://captain.your-site-subdomain.your-domain.com>

Open dicom-store app to get link to Orthanc



Login using Orthanc store credentials, i.e.

CODA_DICOM_STORE_DB_USERNAME
CODA_DICOM_STORE_DB_PASSWORD



Confirm that you are able to login with the default authentication credentials, and upload some non-sensitive DICOMs.

There are many free datasets available, e.g.

<https://cloud.google.com/healthcare-api/docs/resources/public-datasets/nih-chest>

11. Test multi-modal FL

Open the **site** CapRover control panel

<https://captain.your-site-subdomain.your-domain.com>

Predict death from age, gender and chest X-ray



Home Stats Learning Active Connections API

10/26/2023

logout

Fr

Learning

Launch a federated learning model.

Prepare Request

```
[  
    "options": {  
        "inputs": [  
            "gender",  
            "age",  
            "chestXR"  
        ],  
        "outputs": [  
            "isDeceased"  
        ],  
        "optimizer": {  
            "name": "adam",  
            "parameters": {  
                "learning_rate": 0.00004,  
                "validation_split": 0.33,  
                "evaluation_split": 0.2,  
                "epochs": 1,  
                "batch_size": 16,  
                "shuffle": 200  
            }  
        }  
    },  
    "sites": [  
        "112",  
        "111",  
        "113"  
    ]  
]
```

Send Prepare Request

Data Count

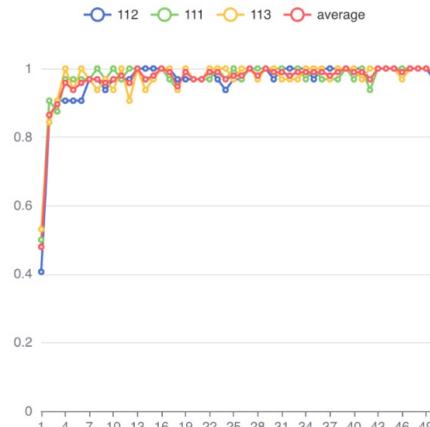
Site Code	Count
-----------	-------

113	100
-----	-----

112	100
-----	-----

111	100
-----	-----

Accuracy



Loss



Input configuration (CODA API syntax)

```
{  
  "selectors": [  
    {  
      "resource": "Patient",  
      "label": "PA",  
      "limit": 1000,  
      "filters": [],  
      "fields": [  
        {  
          "path": "age",  
          "label": "age",  
          "type": "integer"  
        },  
        {  
          "path": "isDeceased",  
          "label": "isDeceased",  
          "type": "boolean"  
        }  
      ]  
    },  
    {  
      "resource": "Observation",  
      "label": "OB",  
      "filters": [  
        {  
          "path": "code.coding.code",  
          "operator": "is",  
          "value": "20570-8",  
          "type": "string"  
        }  
      ],  
      "fields": [  
        {  
          "path": "value.Quantity.value",  
          "label": "hematocrit",  
          "type": "integer"  
        }  
      ]  
    }  
  ]  
}
```

Model configuration (Keras JSON syntax)

```
{  
  "options": {  
    "model": {  
      "class_name": "Sequential",  
      "config": {  
        "name": "sequential_1",  
        "layers": [  
          "etc..."  
        ]  
      }  
    },  
    "inputs": [  
      "gender",  
      "age",  
      "hematocrit"  
    ],  
    "outputs": [  
      "isDeceased"  
    ],  
    "optimizer": {  
      "name": "adam",  
      "parameters": {  
        "learning_rate": 0.00025,  
        "validation_split": 0.33,  
        "evaluation_split": 0.2,  
        "epochs": 1,  
        "batch_size": 20,  
        "shuffle": 1000  
      }  
    },  
    "compiler": {  
      "parameters": {  
        "loss": "binaryCrossentropy",  
        "metrics": [  
          "accuracy"  
        ]  
      }  
    }  
  }  
}
```

12. Test interactive notebooks

Open the **hub** CapRover control panel

<https://captain.your-hub-subdomain.your-domain.com>

1. Open notebook-app in the hub CapRover and click on the link to open JupyterLab

The screenshot shows the 'notebook-app' configuration page on CapRover. It has tabs for 'HTTP Settings', 'App Configs', and 'Deployment'. Under 'HTTP Settings', it says 'Your app is internally available as `srv-captain--notebook-app`' to other CapRover services. There is a checkbox for 'Do not expose as web-app'. Below that, it says 'Your app is publicly available at:' and shows a button to 'Enable HTTPS' with the URL <http://notebook-app.hub-sandbox.coda-platform.com>.

2. Install the coda_api Python library inside the notebook and fetch an access token

```
[3]: %capture  
!pip install coda_api  
  
[5]: from coda_api import hub  
import pprint  
  
access_token = hub.get_access_token({  
    'username': 'test-user',  
    'password': 'unsafeunsafe',  
    'client_id': 'notebook-app',  
    'client_secret': '3bqaJltl87012qH6ccDnKXhYlfQYqCgX',  
    'grant_type': 'password'  
})
```

3. Use the coda_api to query the hub from the Jupyter notebook environment

```
[6]: query = {  
    "selectors": [  
        {  
            "resource": "Patient",  
            "label": "Patient_0",  
            "fields": [  
                { "path": "age",  
                  "label": "patient_age",  
                  "type": "integer" }  
            ]  
        }  
    ],  
    "options": {  
        "measures": {  
            "continuous": [ "count", "mean", "stdev" ],  
            "categorical": []  
        }  
    }  
}  
  
sites = ['111']  
data = hub.execute_query('stats', 'summarize',  
                        sites, query, access_token)  
  
pp = pprint.PrettyPrinter()  
pp pprint(data)  
  
[[{'results': [{ 'count': 5810,  
               'field': 'age',  
               'mean': {'mean': 46.854838709677416, 'populationSize': 124},  
               'measure': 'continuous',  
               'stdev': 25.3061557948256}],  
   'siteCode': '111',  
   'total': 124}],  
[{'results': [{ 'count': 5810,  
               'field': 'age',  
               'mean': {'mean': 46.854838709677416, 'populationSize': 124},  
               'measure': 'continuous',  
               'stdev': 25.30569030573808}],  
   'siteCode': 'all'}]
```

For more examples, visit the notebook-app GitHub:
<https://github.com/coda-platform/notebook-app>

Appendix A

Detailed Keycloak setup

Create new realm named “hub”

The screenshot shows the Keycloak administration interface. On the left, a sidebar menu is visible with various management options like Clients, Client Scopes, Roles, and Authentication. The main content area is titled "Hub" and shows a configuration page for a realm. The "General" tab is selected, displaying fields for Name, Display name, HTML Display name, Frontend URL, Enabled status (set to ON), and User-Managed Access (set to OFF). Below these are sections for Endpoints (containing OpenID Endpoint Configuration and SAML 2.0 Identity Provider Metadata) and Save/Cancel buttons. A blue arrow points from the text "CODA_HUB_API_AUTH_REALM" at the bottom of the image to the "Name" input field on the screen.

Hub

General Login Keys Email Themes Localization Cache Tokens Client Registration Client Policies Security Defenses

* Name hub

Display name

HTML Display name

Frontend URL

Enabled ON

User-Managed Access OFF

Endpoints

OpenID Endpoint Configuration

SAML 2.0 Identity Provider Metadata

Save Cancel

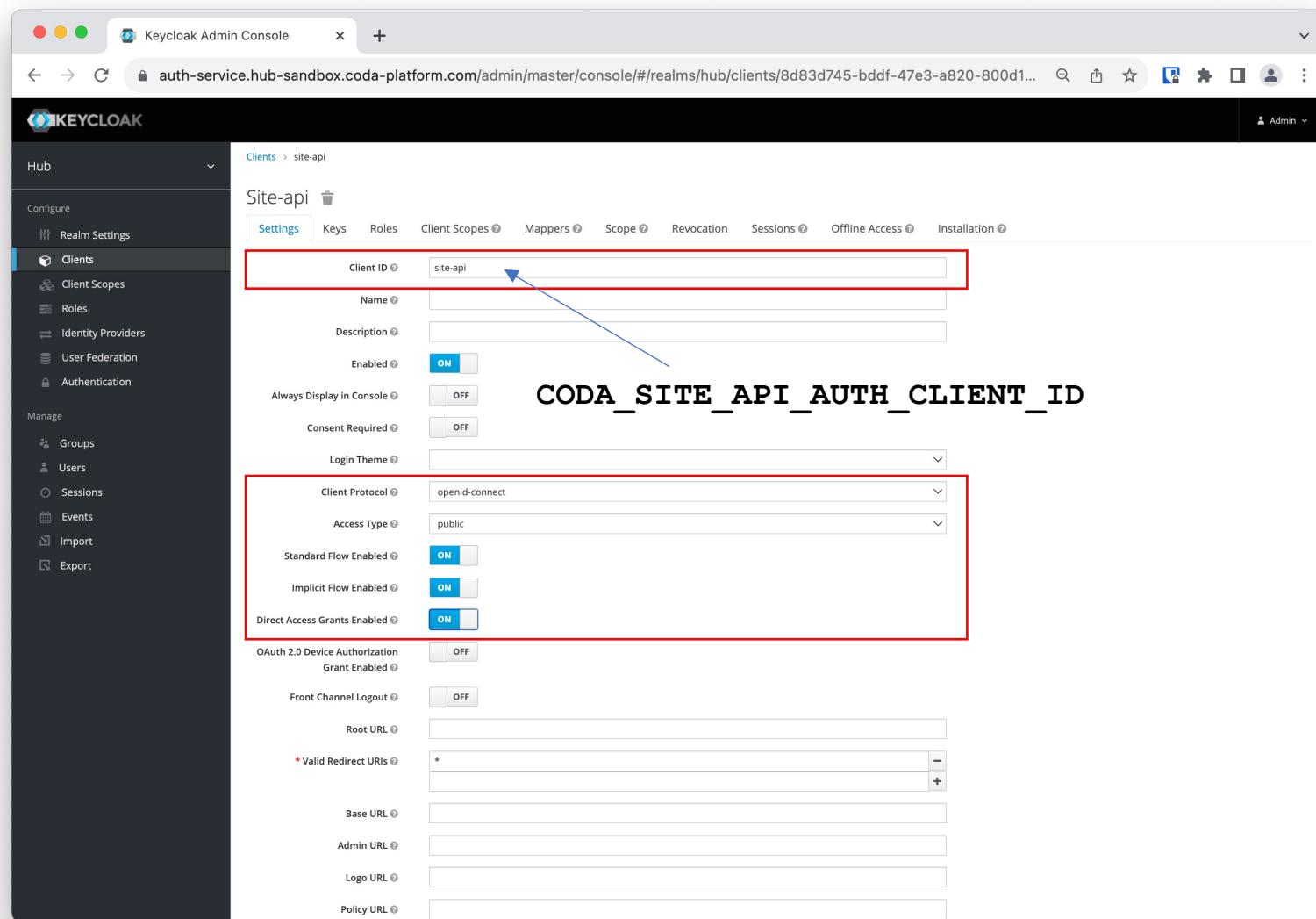
CODA_HUB_API_AUTH_REALM

Create new client named “dashboard-app”

CODA_DASHBOARD_APP_AUTH_CLIENT_ID

The screenshot shows the Keycloak administration interface. On the left, a sidebar menu is open under the 'Hub' section, showing options like 'Clients' (which is selected and highlighted in blue), 'Client Scopes', 'Roles', 'Identity Providers', 'User Federation', and 'Authentication'. The main content area is titled 'Clients > dashboard-app' and shows the 'Settings' tab for the 'Dashboard-app' client. The client ID is set to 'dashboard-app', which is highlighted with a red box. The 'Enabled' switch is turned 'ON', also highlighted with a red box. The 'Access Type' is set to 'public', highlighted with a red box. The 'Standard Flow Enabled' switch is turned 'ON', highlighted with a red box. The 'Valid Redirect URIs' field contains the URL 'https://dashboard-app.hub.coda-platform.com/*', which is highlighted with a red box. The 'Web Origins' field at the bottom is empty.

Create new client named “site-api”



The screenshot shows the Keycloak Admin Console interface. On the left, there's a sidebar with 'Clients' selected. The main area shows a client configuration page for 'Site-api'. A large red box highlights the 'Client ID' field, which contains the value 'site-api'. A blue arrow points from the text 'CODA_SITE_API_AUTH_CLIENT_ID' to this field. Below the 'Client ID' field, another red box highlights a group of protocol-related settings: 'Client Protocol (openid-connect)', 'Access Type (public)', 'Standard Flow Enabled (ON)', 'Implicit Flow Enabled (ON)', and 'Direct Access Grants Enabled (ON)'. The rest of the client configuration page includes fields for 'Name', 'Description', 'Enabled (ON)', 'Always Display in Console (OFF)', 'Consent Required (OFF)', 'Login Theme', 'Root URL', 'Valid Redirect URIs (*, +)', 'Base URL', 'Admin URL', 'Logo URL', and 'Policy URL'.

Create new client named “dashboard-api”

The image shows two screenshots of the Keycloak Admin Console interface. The left screenshot, labeled #1, shows the 'Settings' tab of the 'dashboard-api' client configuration. It highlights the 'Client ID' field (containing 'dashboard-api') and the 'Access Type' dropdown (set to 'bearer-only'). The right screenshot, labeled #2, shows the 'Credentials' tab of the same client, highlighting the 'Secret' field which contains the value '32vBfm2wFyehFN2WK9jHCSANAgWAM'. Blue arrows point from the highlighted fields in the first screenshot to the corresponding fields in the second screenshot.

#1

#2

CODA_DASHBOARD_API_AUTH_CLIENT_ID

CODA_DASHBOARD_API_AUTH_CLIENT_SECRET

Keycloak Admin Console

Dashboard-api

Settings

Client ID: dashboard-api

Name:

Description:

Enabled: ON

Consent Required: OFF

Login Theme:

Client Protocol: openid-connect

Access Type: bearer-only

Front Channel Logout: OFF

Admin URL:

Logo URL:

Policy URL:

Terms of service URL:

Fine Grain OpenID Connect Configuration

OpenID Connect Compatibility Modes

Advanced Settings

Save Cancel

Success! The secret has been changed.

Dashboard-api

Credentials

Client Authenticator: Client Id and Secret

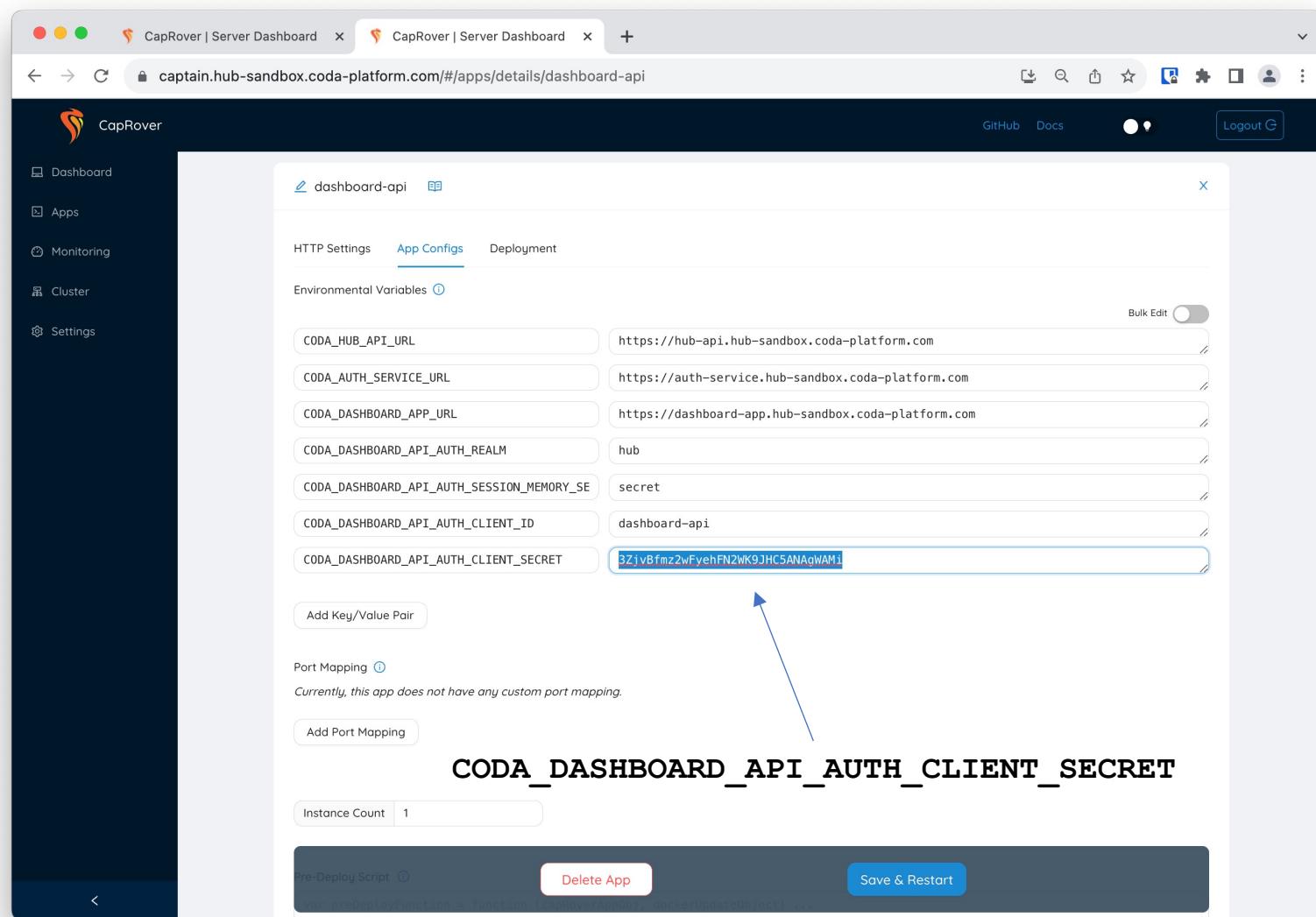
Secret: 32vBfm2wFyehFN2WK9jHCSANAgWAM

Regenerate Secret

Registration access token:

Regenerate registration access token

Update and restart secrets in dashboard-api app



Update to value obtained from Keycloak, then click “Save & Restart”

Create new client named “hub-api”

The image shows two screenshots of the Keycloak Admin Console interface.

Screenshot 1 (Left): Shows the configuration page for the client "hub-api". The "Client ID" field is set to "hub-api". The "Enabled" switch is set to "ON". The "Direct Access Grants Enabled" switch is also set to "ON". Other settings like "Always Display in Console", "Consent Required", and "Implicit Flow Enabled" are set to "OFF".

Screenshot 2 (Right): Shows the configuration page for the client "dashboard-api". The "Secret" field contains the value "32vBfmz2wFyehFN2WK9jHCSANAgWAM". A success message at the top right says "Success! The secret has been changed."

A large blue arrow labeled "#2" points from the "Secret" field in the "dashboard-api" screenshot down to the text "CODA_DASHBOARD_API_AUTH_CLIENT_SECRET" at the bottom of the image.

Create new user named “site1-sandbox”

The image shows two screenshots of the Keycloak administration interface. The left screenshot displays the 'Add user' form, where the 'Username' field is populated with 'site1-sandbox'. The right screenshot shows the 'Manage Credentials' page for the user 'Site1-sandbox', where the 'Password' field is set to 'unsafeunsafe'.

Add user

- ID: [empty]
- Created At: [empty]
- Username ***: site1-sandbox
- Email: [empty]
- First Name: [empty]
- Last Name: [empty]
- User Enabled: ON
- Email Verified: OFF
- Groups: Select existing group... (No group selected)
- Required User Actions: [empty]
- Save | Cancel

Manage Credentials

Position	Type	User Label	Data	Action

Set Password

- Password: unsafeunsafe
- Password Confirmation: unsafeunsafe
- Temporary: OFF
- Set Password

CODA_SITE_API_AUTH_USERNAME
CODA_SITE_API_AUTH_PASSWORD

Create new user named “test-user”

The screenshot shows the Keycloak Admin UI interface. On the left, a dark sidebar menu titled "Hub" lists various management options: Configure (Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication), Manage (Groups, Users, Sessions, Events, Import, Export). The "Users" option under "Manage" is currently selected.

The main content area displays a user profile for "Test-user". The "Details" tab is active. The user's ID is 4e885f89-f795-4eb9-b434-23b0865dedaa, and they were created at 12/21/22 9:56:30 PM. The "Username" field contains "test-user". Other fields like Email, First Name, Last Name, and User Enabled (set to ON) are empty or not applicable. A modal window is open over the details tab, showing "Required User Actions" with an "Impersonate" button and "Save" and "Cancel" buttons.

A blue arrow points from the "Username" field in the main user profile to the "Credentials" tab in the second screenshot.

The second screenshot shows the "Credentials" tab for the "Test-user" profile. It has tabs for Details, Attributes, Credentials (which is active), Role Mappings, Groups, Consents, and Sessions. The "Manage Credentials" section includes a table with columns Position, Type, User Label, and Data. Below it is a "Set Password" form with fields for Password (containing "unsafeunsafe"), Password Confirmation (containing "unsafeunsafe"), and Temporary (set to OFF). A "Set Password" button is at the bottom.

A blue arrow points from the "Password" field in the "Set Password" form to the "Credentials" tab in the main screenshot.

Use these credentials to login to the dashboard.

Appendix B

Interacting with CODA using the REST API

POST {{CODA_AUTH_SERVICE_URL}}/realms/{{CODA_DASHBOARD_API_AUTH_REALM}}/protocol/openid-connect/token

Example request

KEY	VALUE
username	site1-sandbox
password	unsafeunsafe
client_id	site-api
grant_type	password



Test logging in to hub as hospital site

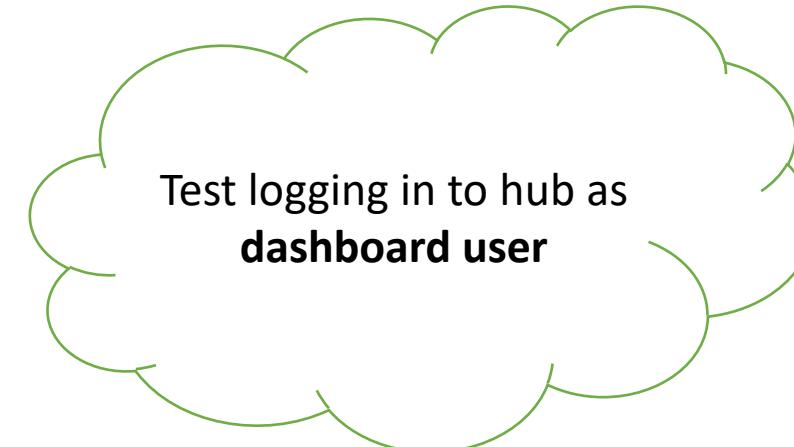
Example response

```
1 ▼ {  
2   "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUExgHhcBBKJe...",  
3   "expires_in": 300,  
4   "refresh_expires_in": 1800,  
5   "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2l...",  
6   "token_type": "Bearer",  
7   "not-before-policy": 0,  
8   "session_state": "378a9517-ac6a-414d-8056-b73c6f0c365a",  
9   "scope": "email profile"  
10 }
```

POST {{CODA_AUTH_SERVICE_URL}}/realms/{{CODA_DASHBOARD_API_AUTH_REALM}}/protocol/openid-connect/token

Example request

KEY	VALUE
username	test-user
password	unsafeunsafe
client_id	dashboard-api
client_secret	vTruOr8IfSV2ljXfVvhi0iMddpEChN1G
grant_type	password



Test logging in to hub as dashboard user

Example response

```
1 ▾ {  
2   "access_token": "eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2l... ",  
3   "expires_in": 300,  
4   "refresh_expires_in": 1800,  
5   "refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2l... ",  
6   "token_type": "Bearer",  
7   "not-before-policy": 0,  
8   "session_state": "661ee647-0bc6-4ddc-891a-cc6c675022cb",  
9   "scope": "email profile"  
10 }
```

```
GET {{CODA_DASHBOARD_API_URL}}/sites
```

Example request

KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Token from response to /openid-connect/token when logging in as dashboard user (previous slide)



Example response

```
1 {  
2   "connections": [  
3     {  
4       "uid": "111",  
5       "name": "Site A Hospital",  
6       "names": {  
7         "fr": "Hôpital Site A",  
8         "en": "Site A Hospital"  
9       },  
10      "api_version": "1.0.1",  
11      "resources": [  
12        {  
13          "resource": "Patient",  
14          "count": {  
15            "estimate": 124  
16          }  
17        },  
18        "more resources here..."  
19      ],  
20      "last_seen": "2023-10-26T10:29:32.363Z"  
21    },  
22    {  
23      "uid": "112",  
24      "name": "Site B Hospital",  
25      "names": {  
26        "fr": "Hôpital Site B",  
27        "en": "Site B Hospital"  
28      }  
29    }  
30  }  
31 }
```

```
GET {{CODA_HUB_API_URL}}/stats/summarize?sites=111,112
```

Example request

KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Example request body

```
1 {  
2   "selectors": [  
3     {  
4       "resource": "Patient",  
5       "label": "patient",  
6       "fields": [  
7         {  
8           "path": "age",  
9           "label": "patient_age",  
10          "type": "integer"  
11        }  
12      ]  
13    }  
14  ],  
15  "options": {  
16    "measures": {  
17      "continuous": [  
18        "count",  
19        "mean",  
20        "stdev"  
21      ],  
22      "categorical": []  
23    }  
24  }  
25}
```

Example response

```
1 [ [ {  
2   "siteCode": "111",  
3   "total": 124,  
4   "results": [  
5     {  
6       "field": "age",  
7       "measure": "continuous",  
8       "count": 5810,  
9       "mean": {  
10         "mean": 46.854838709677416,  
11         "populationSize": 124  
12       },  
13       "stdev": 25.3061557948256  
14     }  
15   ]  
16 }, [ {  
17   "siteCode": "112",  
18   "total": 135,  
19   "results": [  
20     {  
21       "field": "age",  
22       "measure": "continuous",  
23     }  
24   ]  
25 }  
26 ]  
27 ]
```

Test federated analytics functionality

```
GET {{CODA_HUB_API_URL}}/learning/prepare?sites=111,112
```

Example request headers (same as previous)

KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Example request body

```
1 {  
2   "selectors": [  
3     {  
4       "resource": "Patient",  
5       "label": "patient",  
6       "limit": 1000,  
7       "filters": [],  
8       "fields": [  
9         { "path": "age", "label": "age", "type": "integer" },  
10        { "path": "isDeceased", "label": "isDeceased", "type": "boolean" }  
11      ]  
12    }  
13  ],  
14  "options": {  
15    "model": {},  
16    "inputs": ["age"],  
17    "outputs": ["isDeceased"],  
18    "optimizer": {  
19      "name": "adam",  
20      "parameters": { "batch_size": 2, "learning_rate": 0.0001 }  
21    },  
22    "compiler": {  
23      "parameters": {  
24        "loss": "binaryCrossentropy",  
25        "metrics": ["accuracy"]  
26      }  
27    }  
28  }  
29 }
```

Example response

```
1 [  
2   {  
3     "job": "CcVNAu+LC89V/S3D",  
4     "siteCode": "111",  
5     "count": 124  
6   },  
7   {  
8     "job": "CcVNAu+LC89V/S3D",  
9     "siteCode": "112",  
10    "count": 124  
11  }  
12 ]
```

Test preparing a dataset
for a training task

```
GET {{CODA_HUB_API_URL}}/learning/train?sites=111,112
```

Example request headers (same as previous)

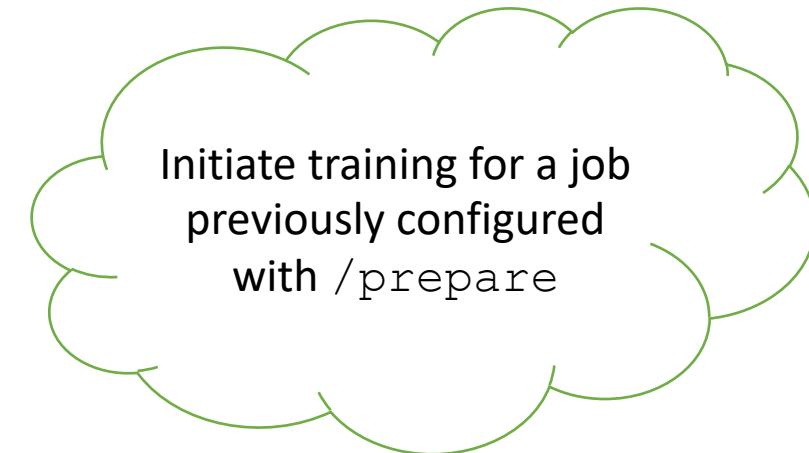
KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Example request body

```
1 {  
2   "job": "FAdmIfw5FxzhiDHJ",  
3   "rounds": 2  
4 }
```

Example request response

```
1 {  
2   "job": "FAdmIfw5FxzhiDHJ",  
3   "rounds": 2  
4 }  
5
```



```
GET {{CODA_HUB_API_URL}}/learning/progress?sites=111,112
```

Example request headers (same as previous)

KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Example request body

```
1 {  
2   "job": "FAdmIfw5FxzhiDHJ"  
3 }
```



Example request response

```
1 [  
2   {  
3     "acc": 0.75,  
4     "loss": 0.6873877048492432,  
5     "val_acc": 0.8064516186714172,  
6     "val_loss": 0.6868967413902283,  
7     "siteCode": "111",  
8     "currentRound": 1,  
9     "totalRounds": 2  
10    },  
11    {  
12      "acc": 0.7983870506286621,  
13      "loss": 0.6893560886383057,  
14      "val_acc": 0.8064516186714172,  
15      "val_loss": 0.6865110993385315,  
16      "siteCode": "112",  
17      "currentRound": 1,  
18      "totalRounds": 2  
19    },  
20    {  
21      "acc": 0.774193525314331,  
22      "loss": 0.6883718967437744,  
23      "val_acc": 0.8064516186714172,  
24      "val_loss": 0.6867039203643799,  
25      "siteCode": "average",  
26      "currentRound": 1,  
27      "totalRounds": 2  
28    }  
29 ]
```

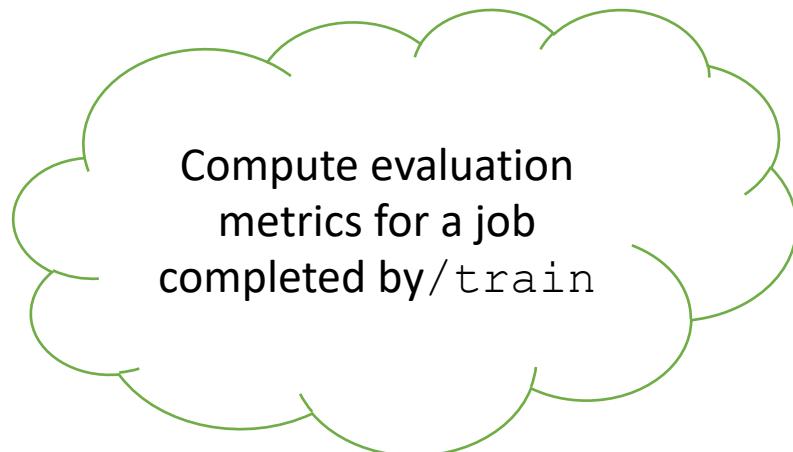
```
GET {{CODA_HUB_API_URL}}/learning/evaluate?sites=111,112
```

Example request headers (same as previous)

KEY	VALUE
Content-Type	application/json
Authorization	Bearer {{access_token}}

Example request body

```
1 {  
2   "job": "FAdmIfw5FxzhiDHJ"  
3 }
```



Example response

```
1 [  
2   {  
3     "siteCode": "average",  
4     "acc": 0.8064516186714172,  
5     "loss": 0.6836749911308289  
6   },  
7   {  
8     "siteCode": "112",  
9     "acc": 0.8064516186714172,  
10    "loss": 0.6825025677680969  
11  },  
12  {  
13    "siteCode": "111",  
14    "acc": 0.8064516186714172,  
15    "loss": 0.6848474144935608  
16  }  
17 ]
```