
KubePortal

Release v0.6.15

Aug 18, 2022

Contents

1	Welcome!	3
2	Installation	9
2.1	Installation with Kustomize	9
2.2	First backend access	10
2.3	Configuration options	10
3	User management	13
3.1	Managing user details	14
3.2	Managing cluster access	14
3.3	Superuser	17
3.4	Merging Users	18
4	User groups	21
4.1	Auto-add	22
4.2	Administrators	22
4.3	Web application access	22
5	Web applications	25
5.1	Frontend link	26
5.2	Login through portal (OIDC)	26
5.3	Login through portal (sub-auth)	27
6	API	29
6.1	First call	29
6.2	Base URL for subsequent calls	29
6.3	Security	29
7	Development Environment	31
7.1	Developing code	31
7.2	Staging test	32
7.3	Logging	32
8	Changelog	33
8.1	v0.3.13 Release	33
8.2	v0.3.12 Release	33
8.3	v0.3.8 Release	33

8.4	v0.3.7 Release	33
8.5	v0.3.6 Release	34
8.6	v0.2.7 Release	34
8.7	v0.2.5 Release	34
8.8	v0.2.4 Release	34
8.9	v0.2.2 Release	34
8.10	v0.2.1 Release	34

Warning: The manuals are work in progress and therefore incomplete. Feel free to help us with a [pull request](#) on [GitHub](#).

CHAPTER 1

Welcome!

KubePortal is a web portal for Kubernetes clusters. It offers the following main features:

- User login with existing credentials (Active Directory, Google, Twitter, ...)
- Landing page with quick links to web applications (Grafana, K8S Dashboard, ...)
- User-friendly process for requesting Kubernetes cluster access
- Self-service download of `kubectl` config file
- Login for web applications through the portal, including group-based access control
- Admin backend

Here are some screenshots:

Data Science Cluster

Status: ~~XXXXXXXXXXXXXXXXXXXX~~ login is **available**

Fig. 1: Portal users can use an existing account to enter the cluster portal.

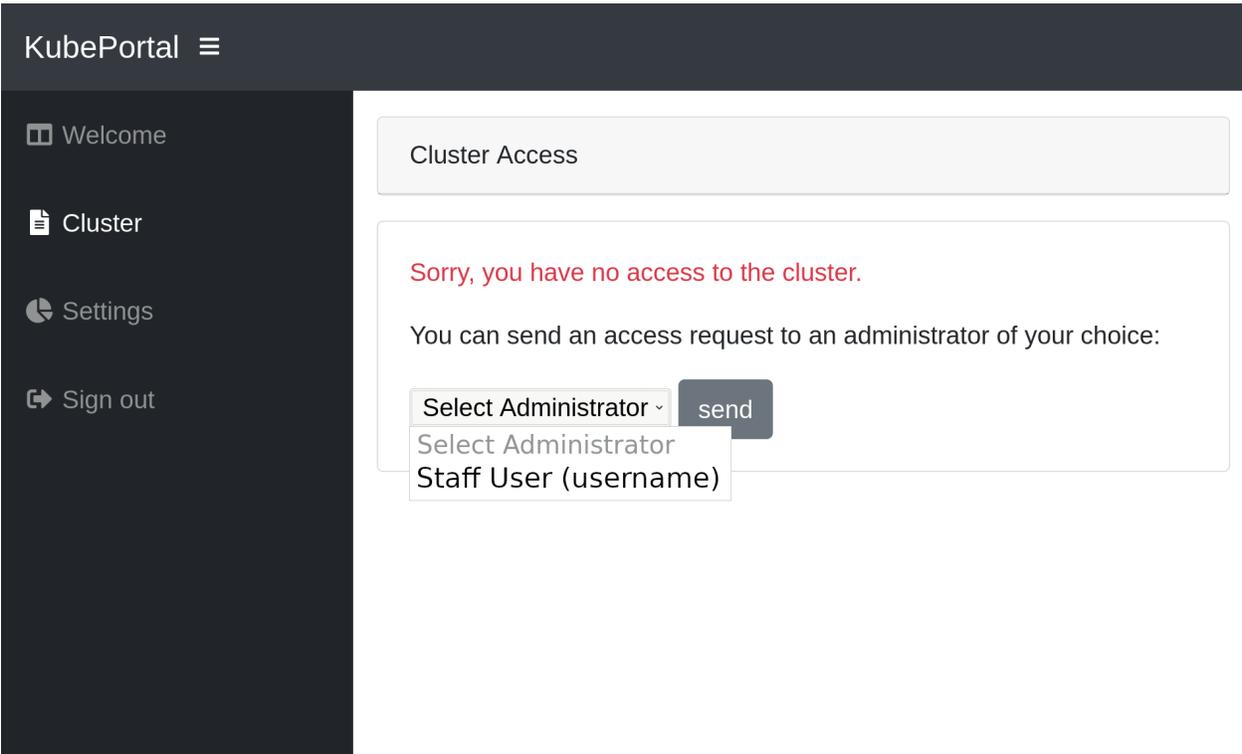


Fig. 2: After login, a landing page is shown that offers the possibility to request Kubernetes access and become a *cluster user*.

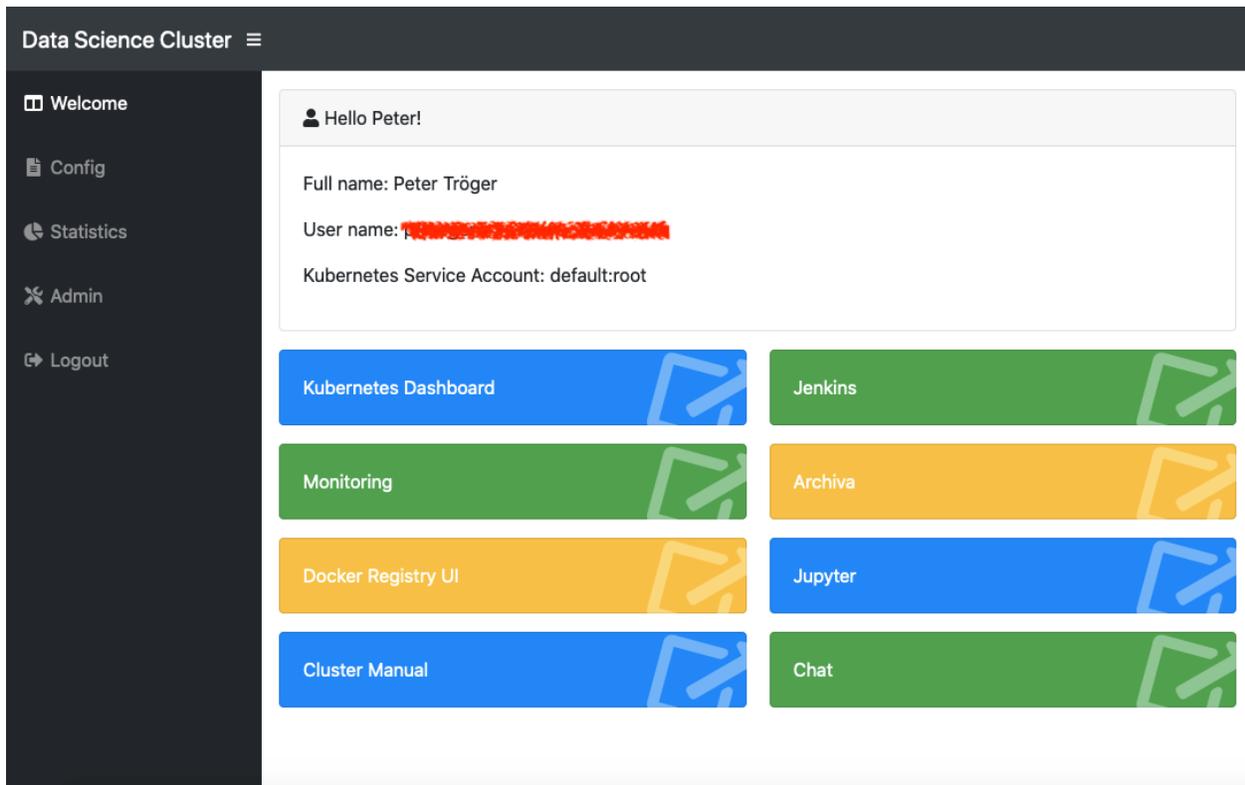


Fig. 3: For approved cluster users, the portal can offer a list of links to other web applications, such as Grafana or Kubernetes Dashboard. The authentication for these web applications is, again, provided through KubePortal, so that users get a single sign-on experience.

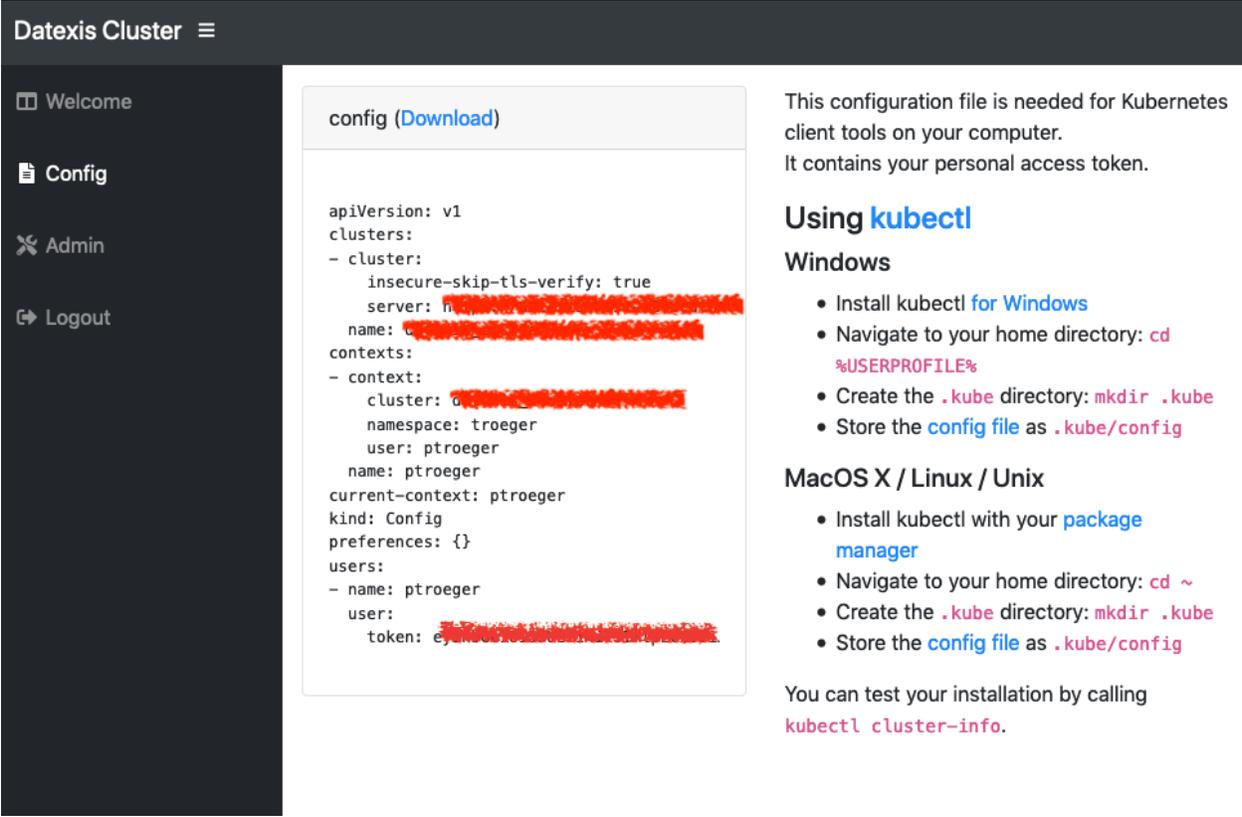


Fig. 4: Approved cluster users also get a download page for their personal KUBECTL configuration file.

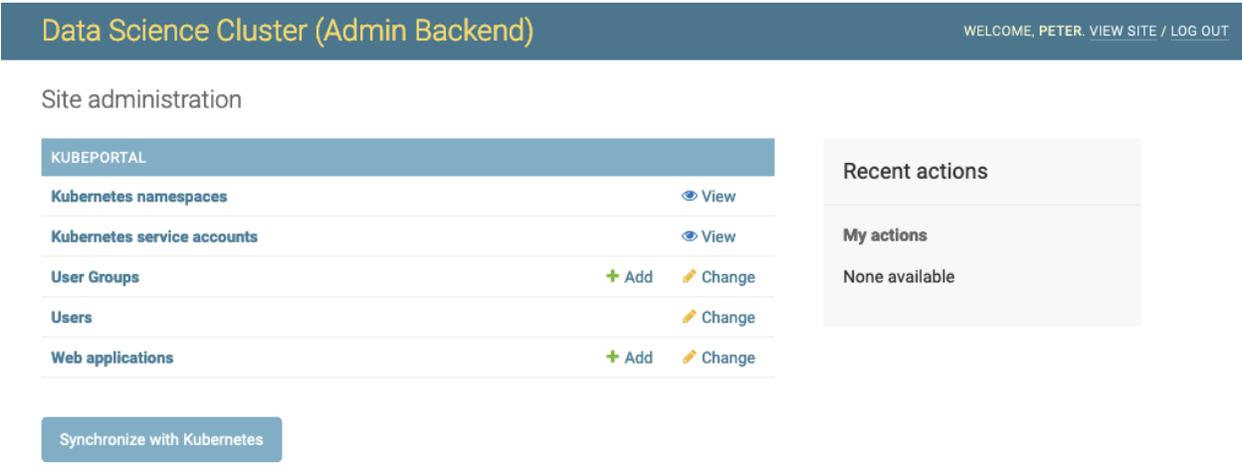


Fig. 5: Portal users with admin rights can access the KubePortal backend, which supports the assignment of known accounts to Kubernetes namespaces resp. service accounts.

The latest official release of KubePortal is always available as [Docker image](#). The software is configured through environment variables (see [Configuration options](#)).

It is *mandatory* to configure the public URLs used by the installation `KUBEPORTAL_ALLOWED_URLS`.

It is *highly recommended* to configure at least one authentication method (`KUBEPORTAL_AUTH_...`) and the database storage.

KubePortal is expected to run inside the Kubernetes cluster it acts as frontend for. The API server is auto-detected from the pod running the software. Permissions must be given to the KubePortal namespace for allowing it to create new namespaces.

2.1 Installation with Kustomize

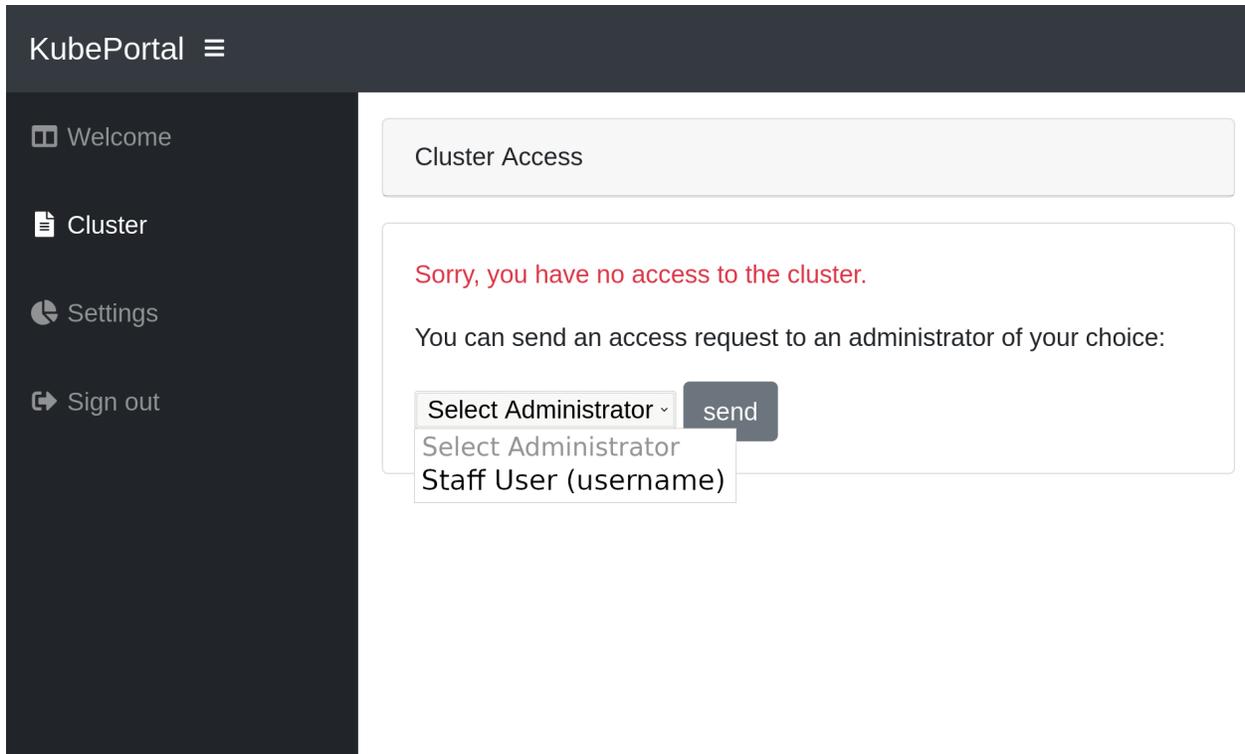
The [source code repository](#) offers Kustomize templates for installation. They perform the following activities:

- A namespace `kubeportal` is created.
- The namespace is configured with the necessary [RBAC permissions](#).
- A persistent volume claim is created.
- A deployment is created:
 - It mounts the persistent volume at `/data`. This allows you to easily configure an SQLite database for storage (`KUBEPORTAL_DATABASE_URL=sqlite:///data/kubeportal.sqlite3`).
 - The environment variables are read from a config map named `kubeportal`. You could create that, for example, from a `.env` file with `kubectl -n kubeportal create configmap kubeportal --from-env-file=.env`.
- A service is created, which makes `kubeportal` available on the service name `kubeportal` at port 8000.

Based on these templates, you could now define your own specialization and apply it to your cluster. Check the Kustomize docs for details about [using remote bases](#).

2.2 First backend access

After installation, first check if your configured frontend authentication method works as expected. A new frontend user should see this welcome screen:



You should now use the superuser login (see Superuser access) to create an admin group (see *User groups*) and add this first user to it.

2.3 Configuration options

Environment variable	Description
KUBEPORTAL_AUTH_TWITTER_KEY	Client key for OAuth when offering frontend Twitter login.
KUBEPORTAL_AUTH_TWITTER_SECRET	Client secret for OAuth when offering frontend Twitter login.
KUBEPORTAL_AUTH_GOOGLE_KEY	Client key for OAuth when offering frontend Google login.
KUBEPORTAL_AUTH_GOOGLE_SECRET	Client secret for OAuth when offering frontend Google login.
KUBEPORTAL_AUTH_GOOGLE_KEY	Client key for OAuth when offering frontend Google login.
KUBEPORTAL_AUTH_GOOGLE_SECRET	Client secret for OAuth when offering frontend Google login.
KUBEPORTAL_AUTH_OIDC_KEY	Client key when offering generic OpenID Connect login.
KUBEPORTAL_AUTH_OIDC_SECRET	Client secret when offering generic OpenID Connect login.
KUBEPORTAL_AUTH_OIDC_ENDPOINT	Endpoint URL when offering generic OpenID Connect login.
KUBEPORTAL_AUTH_OIDC_TITLE	Button title when offering generic OpenID Connect login.
KUBEPORTAL_AUTH_AD_DOMAIN	Domain when offering frontend Active Directory login, e.g. <code>example.com</code> .
KUBEPORTAL_AUTH_AD_SERVER	Active directory server when offering frontend Active Directory login, e.g. <code>19</code>
KUBEPORTAL_API_SERVER_EXTERNAL	URL of the Kubernetes API server that works outside of the cluster, for end us
KUBEPORTAL_SESSION_COOKIE_DOMAIN	The domain used for the user session cookie, e.g. <code>.example.com</code> .
KUBEPORTAL_NAMESPACE_CLUSTERROLES	Kubernetes cluster roles that should be bound to the <i>default</i> service account of

Table 1 – continued from prev

Environment variable	Description
KUBEPORTAL_BRANDING	The human-readable name of your cluster.
KUBEPORTAL_ALLOWED_URLS	The portal URLs used by clients, eg. <code>https://portal.foo.com:8000</code> .
KUBEPORTAL_INGRESS_TLS_ISSUER	The certificate issuer used for Ingress definitions created through the API.
KUBEPORTAL_LANGUAGE_CODE	The locale for the web site, e.g. <code>en-us</code> .
KUBEPORTAL_TIME_ZONE	The time zone for the web site, e.g. <code>UTC</code> .
KUBEPORTAL_ADMIN_NAME	The name of the superuser, used only for email sending.
KUBEPORTAL_ADMIN_EMAIL	The email address of the superuser.
KUBEPORTAL_EMAIL_HOST	The SMTP server used by the web site for sending mails.
KUBEPORTAL_DATABASE_URL	The database to be used as URL (see <i>formatting examples</i> < https://github.com >).
KUBEPORTAL_REDIRECT_HOSTS	Hosts that redirect to the KubePortal web page, typically to perform OAuth authentication.
KUBEPORTAL_ROOT_PASSWORD	The password to be used in the development environment for the <i>root</i> user.
KUBEPORTAL_LOG_LEVEL_PORTAL	Sets the verbosity of the logging for the admin panel. [DEBUG, INFO, WARNING, ERROR]
KUBEPORTAL_LOG_LEVEL_SOCIAL	Sets the verbosity of the logging for <code>django.social</code> . [DEBUG, INFO, WARNING, ERROR]
KUBEPORTAL_LOG_LEVEL_REQUEST	Sets the verbosity of the logging for requests. [DEBUG, INFO, WARNING, ERROR]
KUBEPORTAL_LAST_LOGIN_MONTHS_AGO	Sets how many months ago users have logged in to be considered old in the admin panel.

User management

All users that ever logged into the portal are shown in the user section of the backend:

Data Science Cluster (Admin Backend) WELCOME, PETER. VIEW SITE / LOG OUT

Home > Kubeportal > Users

Select user to change

Q Search

Action: Go 0 of 71 selected

<input type="checkbox"/>	USERNAME	FIRST NAME	LAST NAME	EMAIL ADDRESS	COMMENTS	GROUPS	CLUSTER ACCESS	APPROVED BY	ACTION
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]		All users	not requested	-	-
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	All users, Chat users	not requested	-	-
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	All users, Chat users	rejected	-	-
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]		All users	not requested	-	-
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]	All users, Kubernetes users, Chat users, Demo users	approved	ptroeger	-
<input type="checkbox"/>	[REDACTED]	[REDACTED]	[REDACTED]	[REDACTED]		All users, Chat users	not requested	-	-
<input type="checkbox"/>	afigueroa	Alexei	Figueroa Rosero	Alexei.FigueroaRosero@beuth-hochschule.de	DATEXIS PHD Student	All users, Kubernetes	approved	bwinter	-

A new user has, by default, no access rights to the Kubernetes cluster. She may already be part of one or more user groups (see *User groups*), which could already provide access to chosen web applications. This depends on the particular installation. With a fresh installation, however, new users have no rights for any kind of resource.

3.1 Managing user details

Many of the user details are already taken from the frontend login mechanism, such as the real name or the email address from Active Directory.

The comment field allows to store arbitrary information that is only visible in the backend.

The cluster access status field can be modified directly, but should normally only change as part of an approval procedure (see [Managing cluster access](#)). When users should be removed from cluster access, but not from the portal, it is the easiest to set their *cluster access* status manually to *rejected* here.

Users can belong to a set of user groups, which can be configured individually:

The screenshot displays a user management form with the following sections:

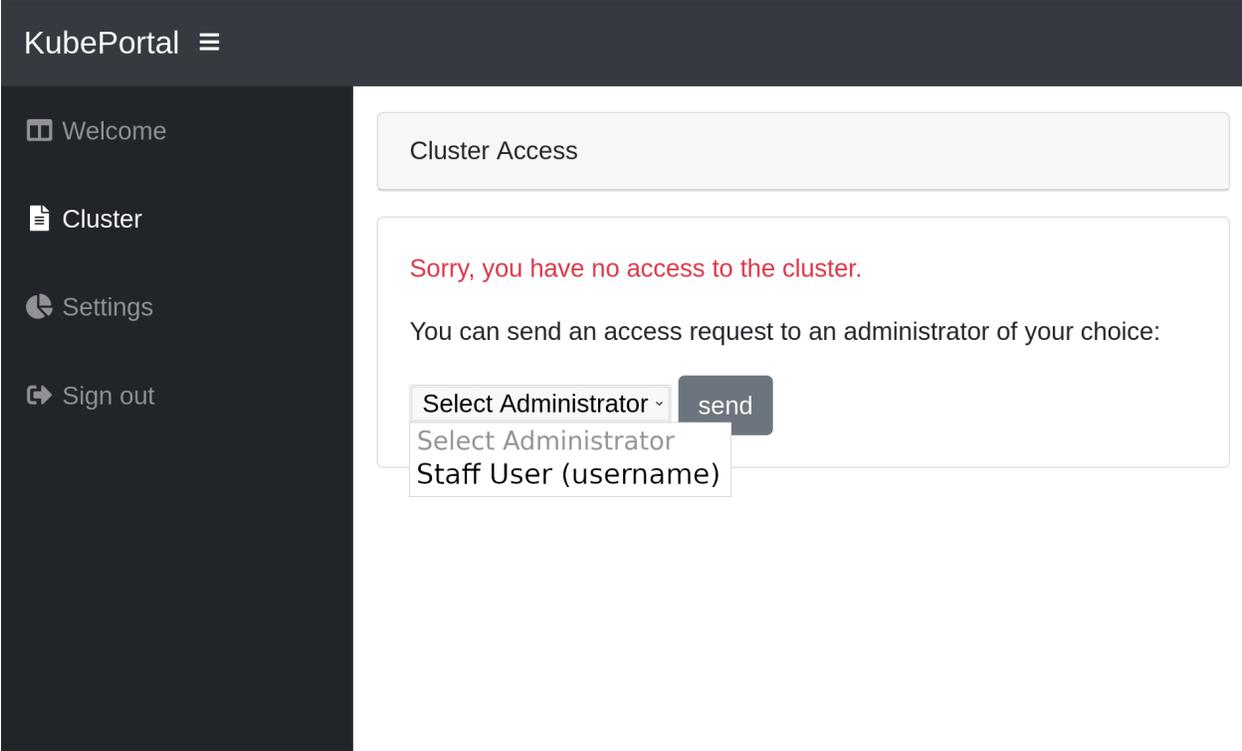
- Kubernetes account:** A dropdown menu with a placeholder "-----" and a small downward arrow. Below it, the text reads "Kubernetes namespace + service account of this user."
- Superuser status:** A red plus icon in a circle. Below it, the text reads "Designates that this user has all permissions without explicitly assigning them."
- Groups:** A section with two panels:
 - Available user groups:** A search box with a magnifying glass icon. Below it, a list of groups: "Kubernetes users", "Administrators", and "Demo users". At the bottom of this panel is a "Choose all" button.
 - Chosen user groups:** A blue header with a plus icon. Below it, a list of selected groups: "All users" and "Chat users". At the bottom of this panel is a "Remove all" button.

Below the groups section, a note states: "The user groups this account belongs to. Hold down 'Control', or 'Command' on a Mac, to select more than one." At the bottom of the form, there are three buttons: "Delete" (red), "Save and continue editing" (blue), and "SAVE" (blue).

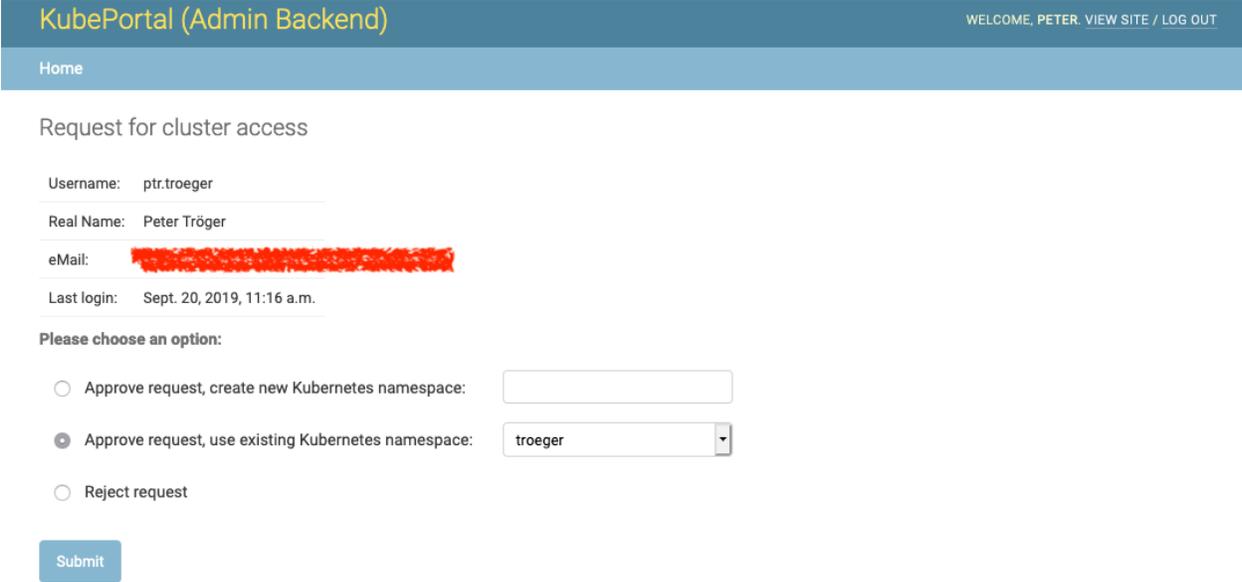
This gives the user permissions for web applications or the backend itself (see [User groups](#)). You can manually add single users to groups by editing their user details directly. Alternatively, it is also possible to perform bulk addition to groups on the user overview page.

3.2 Managing cluster access

Users can click a link on the front page to apply for Kubernetes credentials:



When the user logged in, he can select a staff user from a dropdown menu, to request cluster access. The selected staff user will be notified by email about the request. The approving or rejecting admin for a user approval request is logged in the database. The link to a decision page is sent with the email. The admin(s) can now decide upon this request:



You have the choice between creating a new Kubernetes namespace for this user, using an existing one, or rejecting the request. KubePortal synchronizes the decision with the Kubernetes API server, so that namespaces are automatically created when needed.

After acknowledging the request, the frontend changes immediately for the portal user. She can now access the *kubectl* config file for the *default* service account in the chosen namespace:

This is needed once after the installation, so that KubePortal gets the initial list of namespaces. It is also necessary when you create or modify namespaces directly in the cluster.

KubePortal provides a basic set of information about your cluster users, so that you can identify dead accounts. These informations are shown on the namespace overview page.

Data Science Cluster (Admin Backend) WELCOME, PETER. [VIEW SITE](#) / [LOG OUT](#)

[Home](#) · [Kubeportal](#) · [Kubernetes namespaces](#)

Select kubernetes namespace to view

Action: 0 of 84 selected

<input type="checkbox"/>	NAME	VISIBLE	PORTAL USERS	CREATED IN KUBERNETES	NUMBER OF PODS
<input type="checkbox"/>	...	✓	...	Feb. 4, 2020, 10:58 a.m.	1
<input type="checkbox"/>	...	✓	...	March 14, 2019, 9 a.m.	0
<input type="checkbox"/>	...	✓	...	April 1, 2020, 2:53 p.m.	0
<input type="checkbox"/>	...	✓	...	Jan. 8, 2020, 4:31 p.m.	0
<input type="checkbox"/>	...	✓	...	Feb. 4, 2020, 10:59 a.m.	2
<input type="checkbox"/>	...	✓	...	Jan. 16, 2019, 8:33 a.m.	0
<input type="checkbox"/>	...	✓	...	Dec. 11, 2019, 2:40 p.m.	0
<input type="checkbox"/>	...	✓	...	May 28, 2019, 12:05 p.m.	14
<input type="checkbox"/>	...	✓	...	Dec. 6, 2019, 8:55 a.m.	0
<input type="checkbox"/>	...	✓	...	Jan. 15, 2019, 3:01 p.m.	0
<input type="checkbox"/>	...	✓	...	Jan. 16, 2019, 8:33 a.m.	0
<input type="checkbox"/>	...	✓	...	Jan. 15, 2019, 3:01 p.m.	12
<input type="checkbox"/>	bwinter	✓	bwinter	Jan. 15, 2019, 3:01 p.m.	9

FILTER

By visible

All

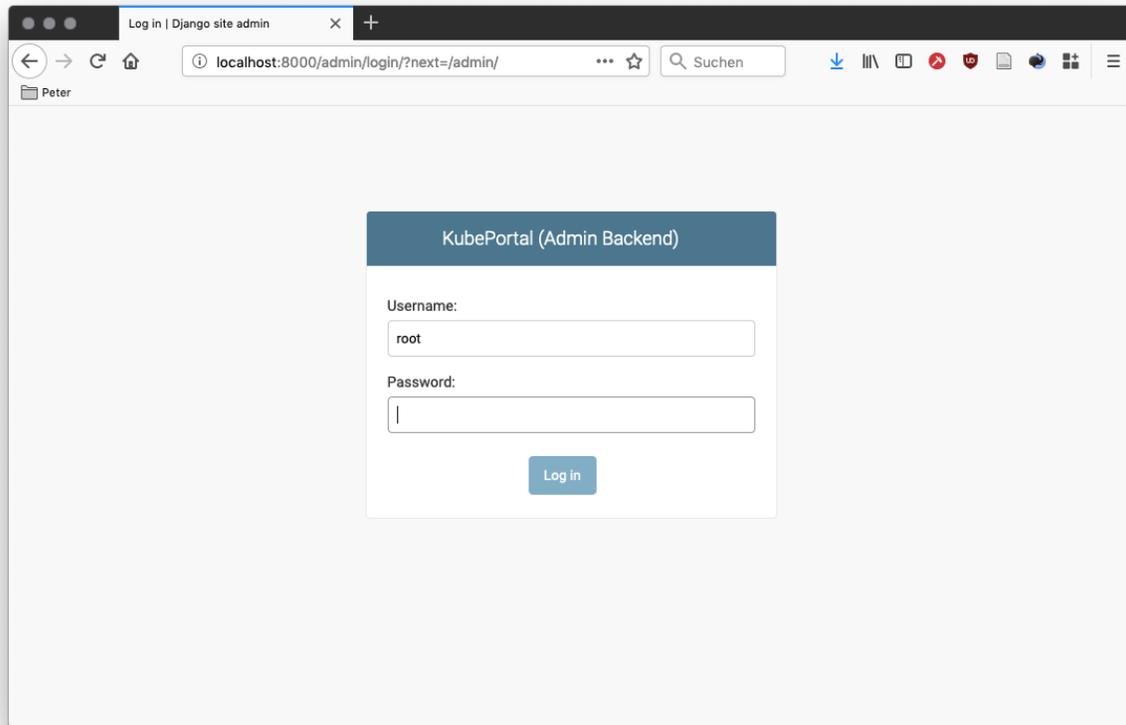
Yes

No

On the overview page, namespaces can be configured for being invisible in the portal. This allows you to hide them from the selection of available namespaces in the approval step above, so that things such as kube-system are not a valid choice for end user accounts.

3.3 Superuser

On startup, the log output of the KubePortal pod shows you a generated password for the root account. This account **only** works on a special backend login page, which is available at <KubePortal URL>/admin/:



It allows you to enter the admin backend while by-passing all configured frontend login methods. This is especially helpful with an empty database after installation.

3.4 Merging Users

In the backend user administration, one is able to select two users and then click the `Merge two users` action. This will merge all the important credentials of the newer (referred to as secondary) into the older (referred to as primary).

Select user to change

ADD USER +

Q Search

Action: Go 2 of 2 selected

-
- Delete selected users
- Reject access request for selected users
- Merge two users
- Assign to group 'All users'
- Assign to group 'Kubernetes users'
- Assign to group 'asdf'

U	MENTS	GROUPS	CLUSTER ACCESS	APPROVED BY	ACTION
<input checked="" type="checkbox"/>			not requested	-	-
<input checked="" type="checkbox"/>			not requested	-	-

In case the secondary user has gotten their cluster access rejected, this will carry over to the primary. Also the user comments from the secondary user will be favored unless the secondary has no comments. Any group that secondary has joined will be carried over to the primary. The secondary user will be deleted after all the details have been merged.

CHAPTER 4

User groups

Portal users are organized into groups. The membership in group decides upon the access rights to the portal backend and web applications.

You can see the list of existing user groups on the group overview page:

Data Science Cluster (Admin Backend) WELCOME, PETER. VIEW SITE / LOG OUT

Home · Kubeportal · User Groups

Select User Group to change ADD USER GROUP +

Action: Go 0 of 5 selected

<input type="checkbox"/>	NAME	MEMBERS	AUTO-ADD NEW USERS	AUTO-ADD APPROVED USERS	BACKEND ACCESS	CAN USE
<input type="checkbox"/>	Demo users	[REDACTED]	✗	✗	✗	CDV Demo
<input type="checkbox"/>	Administrators	[REDACTED]	✗	✗	✓	
<input type="checkbox"/>	Chat users	[REDACTED]	✗	✓	✗	RIS Chat
<input type="checkbox"/>	Kubernetes users	[REDACTED]	✗	✓	✗	K8S Dashboard,

A click on the name brings you to the details page:

Data Science Cluster (Admin Backend)
WELCOME, PETER. [VIEW SITE](#) / [LOG OUT](#)

Home › Kubeportal › User Groups › Demo users

Change User Group HISTORY

Name:

Auto-add new users
Enabling this makes all newly created users automatically a member of this group. Existing users are not modified.

Auto-add approved users
Enabling this makes all newly approved Kubernetes users automatically a member of this group. Existing users are not modified. Users with cluster approval being removed stay in this group.

Backend access
Enabling this allows members of this group to access the administrative backend.

Web applications:

K8S Dashboard
 Grafana
 Docker Registry UI
 Cluster Manual
 Jenkins
 Archiva
 JupyterHub
 RIS Chat
 CDV Demo

+

Web applications that are accessible for members of this group. Hold down "Control", or "Command" on a Mac, to select more than one.

Delete
Save and add another
Save and continue editing
SAVE

4.1 Auto-add

Groups can be configured to be automatically filled with all portal users (“Auto-add new users”) or with all approved cluster users (“Auto-add approved users”). This allows you to give access to cluster-related web applications, such as Grafana or K8S dashboard, only to people that actually got a Kubernetes account.

4.2 Administrators

A user can be granted admin rights by making it member of a group with *Backend access* enabled. It is therefore reasonable to have at least one group where this flag is set, and add yourself to it.

4.3 Web application access

The group membership of a user decides which web applications are accessible for her. These permissions are cumulative, so membership in multiple groups can give user access to different web applications.

The group concept allows you to implement different scenarios:

- All portal users should be able to use a particular web application, regardless of their cluster approval status:
 - Create a group with “auto-add new users” enabled.
 - Give that group access to the web application.
- All users with Kubernetes cluster access should be able to use a particular web application, e.g. Kubernetes Dashboard:

- Create a group with “auto-add approved users” enabled.
 - Give that group access to the Kubernetes Dashboard web application.
- Administrators should have web applications available that are not for ordinary cluster users, e.g. Grafana:
 - Create a group with “Backend access” enabled.
 - Manage the group members manually.
 - Give that group access to the Grafana web application.
- Some web applications should be only available to special people:
 - Create a group and manage the members manually.
 - Give that group access to the chosen web application.

Web applications

KubePortal can be used to manage the access to web applications, which typically run in the cluster itself. A typical example is [Kubernetes Dashboard](#), which can be used as graphical frontend for cluster management.

The current list of configured web applications is accessible in the backend. When creating a new web application entry, you have a set of options for the integration in the portal described below.

Change web application

[HISTORY](#)

Name:

Portal frontend

Show link

Show link on the landing page when user has access rights.

Link title:

The title of the link on the landing page. You can use the placeholders `{{namespace}}` and `{{serviceaccount}}`.

Link URL: **Currently:**

Change:

The URL of the link on the landing page. You can use the placeholders `{{namespace}}` and `{{serviceaccount}}`.

Security

OpenID Connect Client:   

Enable sub-authentication URL

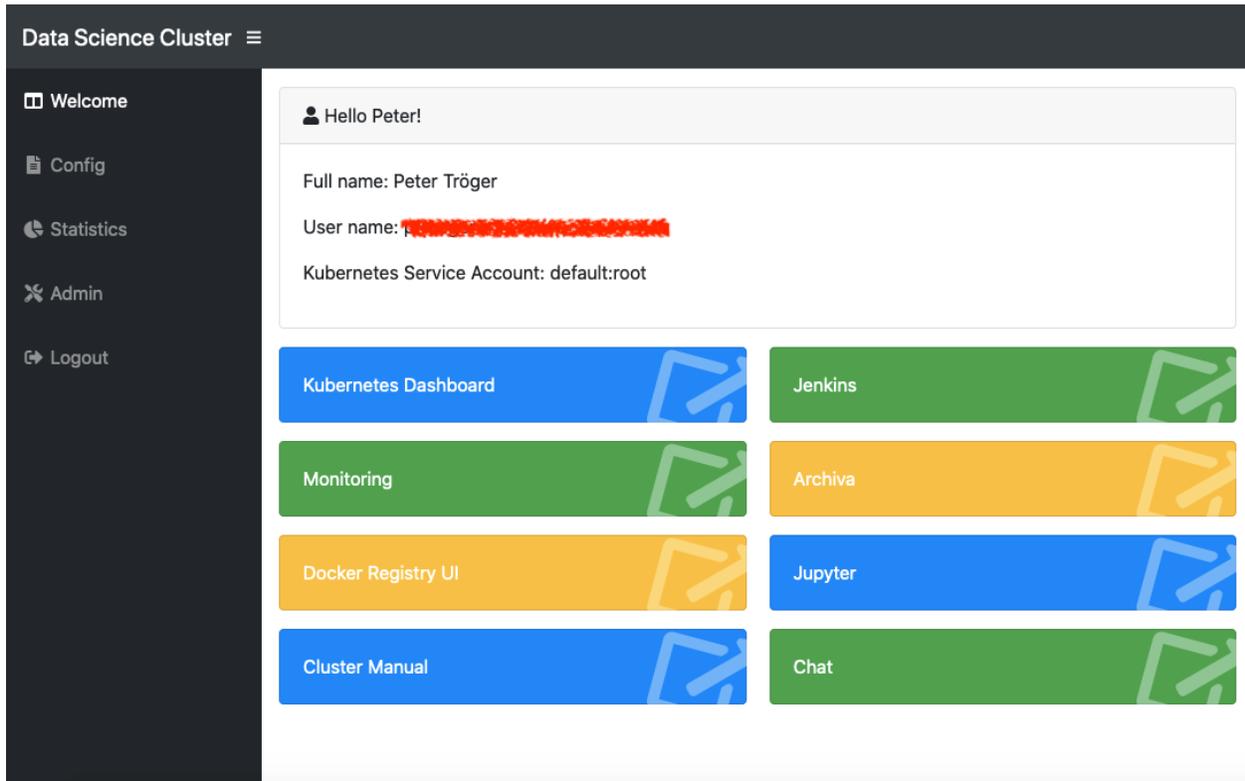
Enables an URL to allow proxy sub-authentication for this web application.

[Delete](#)[Save and add another](#)[Save and continue editing](#)[SAVE](#)

5.1 Frontend link

A web application can be presented as link on the landing page for users. Both the name shown and the URL are customizable. The target URL can use placeholders, so that customized URLs are possible:

- `{{namespace}}`: Inserts the configured Kubernetes namespace name for the portal user in the URL.
- `{{serviceaccount}}`: Inserts the configured Kubernetes service account for the portal user in the URL.



A simple example where this becomes handy is a link to the K8S dashboard page of the user accessing the portal:

```
https://dashboard.example.com/#!/overview?namespace={{namespace}}
```

A link to the web application is only shown when the user has access permissions (see *User groups*).

5.2 Login through portal (OIDC)

KubePortal operates as **OpenID Connect (OIDC)** provider, so web application such as **Grafana** can use the portal for authentication. For users already logged into the portal with the same browser, the web application then just works. Anonymous users are redirected by the web application to the portal for login.

To allow OIDC-enabled web applications to use the portal, you need to create an OpenID Connect client in the web application settings:

Please note that the client secret is generated automatically and can be seen in the web application overview list. The redirect URIs depend on the particular web application you are trying to integrate.

The web application also needs some settings:

- Client ID and client secret are shown in the web application overview page.

Change Client

Client ID:	<input type="text" value="rocket-chat"/>
Redirect URIs:	<div style="border: 1px solid #ccc; padding: 5px; min-height: 80px;">  </div> <p style="font-size: small; margin-top: 5px;">Enter each URI on a new line.</p>
Client SECRET:	ea9140ed648de3e1cc64a70d5b0ed9c1ef8ae8dd5389c71cdc6cf743f6

SAVE

- The recommended authentication scopes are *openid*, *profile*, and *email*.
- The link `<KubePortal URL>/oidc/authorize` provides the authorization endpoint.
- The link `<KubePortal URL>/oidc/token` provides the token information endpoint.
- The link `<KubePortal URL>/oidc/userinfo` provides the API endpoint for fetching user information.
- The returned JSON with user information contains the keys 'nickname' (aka username), 'given_name', 'family_name', 'name', and 'email'.

The login through OIDC is only possible if the user has access permissions for this web application (see *User groups*).

Example configuration for Grafana:

5.3 Login through portal (sub-auth)

When you operate your web applications in the cluster itself, and rely on the [NGINX Ingress Controller](#), it is possible to use another portal authentication method called 'sub-authentication'. One use case for this is the [Kubernetes Dashboard](#).

Sub-authentication can be separately enabled per web application. This generates a special sub-authentication URL that is only valid for this app. This URL can be seen on the web application overview page.

The URL must be added to the Ingress definition of the application about to be protected. The details are described in the [NginX ingress documentation](#).

Example:

Please note that `KUBEPORTAL_SESSION_COOKIE_DOMAIN` (see *Configuration options*) must be set to a value that matches both to your portal and web application DNS name, e.g. `.example.com`, otherwise the login check will

always fail. This means that all web application URLs using this mechanism must live in the same DNS zone as your portal installation.

The login through sub-authentication is only possible if the user has access permissions for this web application (see *User groups*).

KubePortal provides an API for working with information stored in the portal database. The interactive API documentation is available at </api/docs>.

6.1 First call

The initial call should be a *GET* call to the */api/* endpoint of your installation. It provides the CSRF token that is necessary for subsequent requests, and the default API version currently supported by the installation:

```
{
  "csrf_token": "oZmEHPUxOn4VpP7hDpvJFxFx4Y1z8A7JY7ImNsve8r0f8TntUrw1PktuR4Xc1YQAgF",
  "portal_version": "v0.4.0",
  "default_api_version": "v1.4.0"
}
```

The CSRF token must be sent in the *X-CSRFToken* HTTP header of subsequent POST / PATCH / PUT requests. It is not needed for GET requests:

```
X-CSRFToken: oZmEHPUxOn4VpP7hDpvJFxFx4Y1z8A7JY7ImNsve8r0f8TntUrw1PktuR4Xc1YQAgF
```

6.2 Base URL for subsequent calls

The base URL for a given API version is */api/<version>/*, so for the API version *v1.4.0*, the login method is available at */api/v1.4.0/login/*.

6.3 Security

For most operations, authentication is needed. This is realized by providing credentials to the *login/* endpoint. The response contains a JSON dictionary, which offers the authentication token under the *access_token* key. Put the value

into an authorization header field in subsequent request:

Response from *POST /api/<version>/login/* call with credentials:

```
{
  "id": 1,
  "firstname": "",
  "access_token": "difhwoefhjwsefw"
}
```

HTTP header in subsequent requests:

```
Authorization: Bearer difhwoefhjwsefw
```

CORS headers are sent by this application, but the code behaves differently in debug mode and production mode. In the latter case, it is mandatory to set the `KUBEPORTAL_ALLOWED_URLS` configuration option. Check the server log output for details, in case of problems.

Development Environment

There are two ways to launch the KubePortal development environment. One is for editing code, the other to perform a staging test with the production version.

The following software is needed on your computer:

- Python 3
- Minikube (+ dependencies, such as libvirt or kvm2)
- GNU Make

The development environment will require you to create a .env file containing - at least - the following environment variables.

- KUBEPORTAL_DATABASE_URL (usually `sqlite:///data/kubeportal.sqlite3`)
- KUBEPORTAL_AUTH_GOOGLE_SECRET
- KUBEPORTAL_AUTH_GOOGLE_KEY

Information about the variables can be found [here](#).

Note: If you have set up a minikube instance before using virtualbox you might want to either delete the old instance or set up a new, named instance using `kvm`. Otherwise minikube will refuse to start up.

You can always clean your computer from the running Minikube containers and temporary files with:

```
make clean
```

7.1 Developing code

```
make dev-run
```

This command starts a minikube instance and the Django development server so that you can access the portal page at `http://127.0.0.1:8000`. Please note that Minikube is automatically changing your `kubectl` configuration.

You may want to let your local KubePortal instance talk to an external cluster, f.e. to test functionalities against production data. This is trivially possible by activating a different `kubectl` configuration. Whatever `kubectl` is talking to, your KubePortal instance is doing the same.

7.2 Staging test

```
make staging-run
```

This command:

- Starts a minikube instance
- Builds the production Docker image of KubePortal
- Deploys it to the Minikube instance
- Runs a port forwarding so that you can access the page at <http://127.0.0.1:8000>
- It will use the environment variables defined in the `.env` file.

Note: You can check your current container status / logs not only via `kubectl`, but also via the kubernetes dashboard. Just run `minikube dashboard`.

7.3 Logging

Developers are able to set different log levels for KubePortal. By default, it will log as verbose as possible. You change the verbosity using the `KUBEPORTAL_LOG_LEVEL_*` environment variables. Please check the [installation manual](#)

8.1 v0.3.13 Release

- Fix sync with pre-existing namespaces of the same name in K8S
- Fix sync with illegal namespace names after approval

8.2 v0.3.12 Release

- Allow reverse member management in backend at different places (#61)(#62)
- Fix bug with duplicate web apps being shown
- “All users” and “Kubernetes users” are not automatically created and managed groups

8.3 v0.3.8 Release

- Fix a problem with ‘next=’ parameter forwarding on OIDC login from web applications.
- Fix some small glitches in the user administration backend.
- The documentation now reflects the latest features (web apps, user groups, installation methods).

8.4 v0.3.7 Release

- Fix a problem with existing users not seeing the new portal group and web app admin pages.

8.5 v0.3.6 Release

- User groups and web applications were added as new concept, and replace the former direct OpenID connect handling. Check the documentation for details (#4, #6)
- Backend permissions are now solely handled by group membership (#52).
- The Kubernetes namespace list in the backend now has some usage statistics, in order to find zombie namespaces (#43).
- The Kubernetes namespace list in the backend now supports some bulk actions (#24, #18).
- The welcome page is now mobile-friendly (#46).
- The test coverage was greatly extended (#34).
- Sub-authentication can now be separately enabled / disabled per web application. The overview pages were updated accordingly.
- User names for active directory logins are now converted to lower case.

8.6 v0.2.7 Release

- Fix broken auth providers
- Allow override of auto-detected API server, because generated config file needs the world-visible IP address in production.

8.7 v0.2.5 Release

- Fix backend admin optics, allow change of approving person (#23)

8.8 v0.2.4 Release

- Fixen broken dependencies

8.9 v0.2.2 Release

- Database migrations squashed, outdated dependencies removed - attempt to fix #12
- Show approving person in backend overview list

8.10 v0.2.1 Release

- API access for the portal user list (see *API*)
- Support for storing comments about particular users in the backend (#20)
- Support for fine-grained log level configuration (#37)
- eMail address is shown in user backend (#21)

- Kubernetes API server is now automatically detected
- Portal shows some generic statistics about the cluster
- Generic support for OIDC login (thanks to [@cultcom](#))