**Cloud-Based Reliability Requirements:**

Cloud computing is the availability of system resources, especially storage and power, etc. on demand. Stormfront is Kubernetes and git-based hosting service for VUE Stormfront and its name is “VSF V1”.

**What is Kubernetes:**

Kubernetes is an open-source container orchestration system for automating software deployment, scaling, and management. It is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available

**Git-based hosting service:**

It is also a cloud-based service platform where developers manage and store their codes**.** It is a version control system, a tool or engine that anyone can download to their device. It's a free and very fast VCS. It **allows you to host projects that use Git**. They have all the functions of the Git engine and various other additional features

**Getting started**

To use the VUE storefront cloud, we first have to register an account and get the configuration file “kubectl.config” which includes all the tools for connection. After that, we get the two namespaces which are:

**demo-storefrontcloud-io -:** It is the product version of your app Storefront.

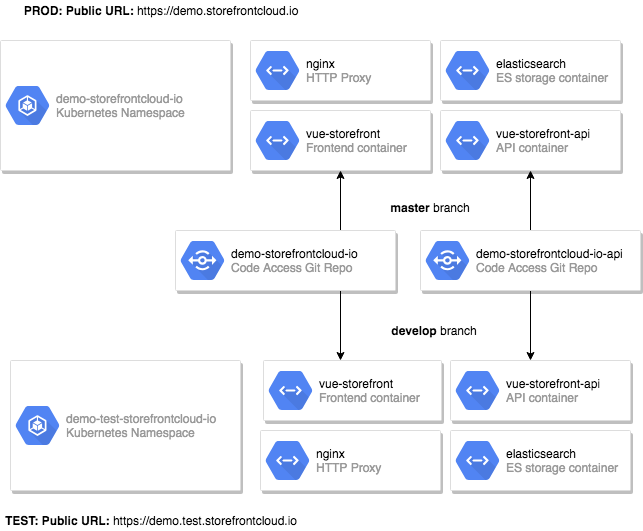
**demo-test-storefrontcloud-io** **-**: It is the test version of your app Storefront.

Below is the diagram in which it was mentioned how the two are set up. After that, you get the git access by Storefront Cloud Site. We get two repositories which are:

**demo-storefrontcloud-io -: Master Branch** for Production frontend and **Develop Branch** for Test frontend.

**demo-storefrontcloud-io-api -: Master Branch** for Production API and **Develop Branch** for Test API.

The namespace we get contains PODs (a container name of Kubernetes):

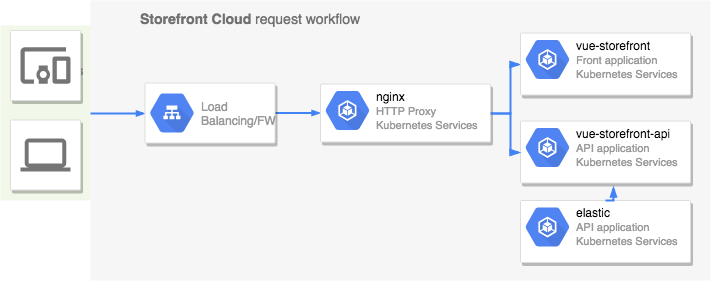
[](https://github.com/StorefrontCloud/storefrontcloud-cli/blob/master/doc/architecture-diagram.png)

**What is POD:**

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes. A Pod (as in a pod of whales or pea pod) is **a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers.**

The namespace is to public addresses which are URL:

* [**demo-storefrontcloud-io**](https://code.storefrontcloud.io/Divante/demo-storefrontcloud-io) : is deployed under [demo.storefrontcloud.io](https://demo.storefrontcloud.io)
* [**demo-storefrontcloud-io-api**](https://code.storefrontcloud.io/Divante/demo-storefrontcloud-io-api) : is deployed under [demo.storefrontcloud.io/api](https://demo.storefrontcloud.io/api)

[](https://github.com/StorefrontCloud/storefrontcloud-cli/blob/master/doc/user-flow-diagram.png)

**Switch POD:**

For switching the POD, we have to write the following command

node scripts/cli.js pod

**The storefront Cloud CLI** tool is designed to manage our Storefront Cloud namespaces. Most of the operations are available for the user.

**Setup:**

After registering the account we get a notification from the team about further details.

* Kube.config file
* Login and password

**Install StoreFront Cloud:**

After that install the storefront cloud. **Clone from the git and the link is given below:**

* + git clone [https://github.com/StorefrontCloud/storefrontcloud-cli.git`](https://github.com/StorefrontCloud/storefrontcloud-cli.git%60)
  + cd storefrontcloud-cli
  + yarn install

**Setup Your account:**

Now run the command:node scripts/cli.js setup

1. Now we have to save files received from the team and some other things as well.
2. Download and setup the kubectl tool if not installed

**Switch namespace:**

We are getting two namespaces:

* **instance-storefrontcloud-io** - which is the main, public instance (accessible via **instance.storefrontcloud.io**),
* **instance-test-storefrontcloud-io** - which is the second development/test, instance (accessible via **instance.test.storefrontcloud.io**).

For switching the namespace, we have to write the following command:

node scripts/cli.js namespace

**Deploy code to POD:**

PODS are configured for code deployment in the initialization process that was provided by the team.

* To deploy the changes to **demo.storefrontcloud.io** You may run:

node scripts/cli.js deploy --pod=front

* To deploy the changes to **demo.storefrontcloud.io/api** You may run:

node scripts/cli.js deploy --pod=api

**Execute the remote command:**

Now we have to execute the remote commands of selected PODs:

node scripts/cli.js exec --pod=front -- ps

node scripts/cli.js exec --pod=api -- ps

**Execute remote shell:**

We have to use the command for executing the remote shell:

node scripts/cli.js exec --pod=api -- sh

**Transfer files and directories between localhost and containers:**

We have to transfer files from one to another by using the following command:

node scripts/cli.js cp api:var/catalog.json catalog-copy.json

This command will copy the file from the container with the role api to a local file named catalog-copy.json.

You can do the reverse operation with the following command:

node scripts/cli.js cp catalog-copy.json api:var/catalog.json

**Make Elastic Search data dump:**

We have to perform the Elastic Search of pumping the data default. Elastic Search data dump will execute the following sequence of commands:

* Dump data on the server,
* Transfer the file using kubectl cp to localhost.

**Restore Elastic Search data dump**

To restore the data dumped with the dump command please run:

node scripts/cli.js restore --input=catalog.json

**Import products from Magento2**

Magento which integrates with the Vue storefront and we have to import data:

node scripts/cli.js import

**Run the PM2 process manager**

Storefront Cloud is using the [pm2](http://pm2.keymetrics.io/) process manager to run both:

vue-storefront

vue-storefront-api.

**Restart the app**

We have to restart the application by running the following command:

node scripts/cli.js pm2 reload all --pod=front

node scripts/cli.js pm2 reload all --pod=api

**Monitor the app**

To monitor the app, we have to use the following command:

node scripts/cli.js pm2 logs server --pod=front

node scripts/cli.js pm2 logs api --pod=api

Cloud-based storefront provide you with many benefits which are:

1. security
2. scalability
3. flexibility

The Code Implementation of Storefront Cloud is given below:

|  |
| --- |
| 1. const KUBECONFIG\_PATH = path.resolve('.kube.config') |
| Variable for Kubernetes Path that we have saved |
|  |
|  |

|  |
| --- |
| 1. const CONFIG\_PATH = path.resolve(‘.storefrontcloud.config’) |
| Variable for Storefront Cloud Path |
|  |

1. const PODS\_CACHE\_PATH=path.resolve('.storefrontcloud.pods.cache')

Variable for Pod Path

|  |
| --- |
| if (fs.existsSync(CONFIG\_PATH)) { |
|  |

|  |
| --- |
| CONTEXT = jsonFile.readFileSync(CONFIG\_PATH) |
|  |

|  |
| --- |
|  |

}

This is basically getting files from the storefront cloud configuration.

|  |
| --- |
| if (fs.existsSync(PODS\_CACHE\_PATH)) { |
|  |

|  |
| --- |
| PODS\_CACHE = jsonFile.readFileSync(PODS\_CACHE\_PATH) |
|  |

}

This is basically loading namespace and POD for configuration

Here is the link from where there is a complete understanding of Code

<https://github.com/StorefrontCloud/storefrontcloud-cli/blob/master/scripts/cli.js>