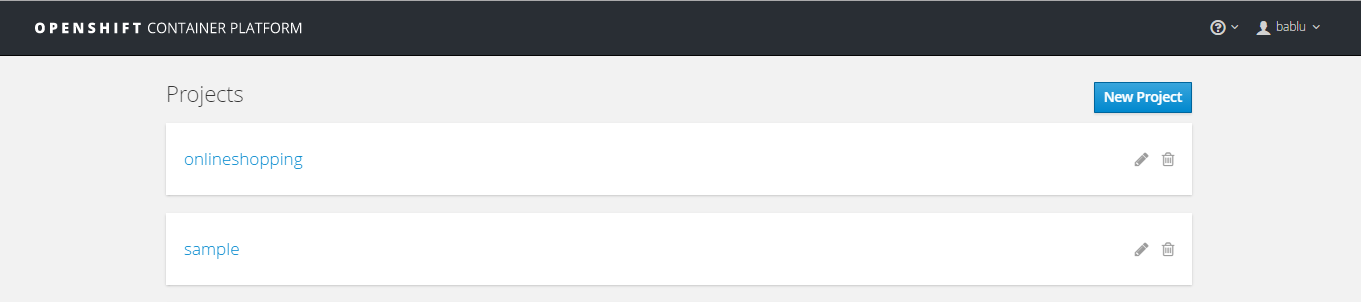
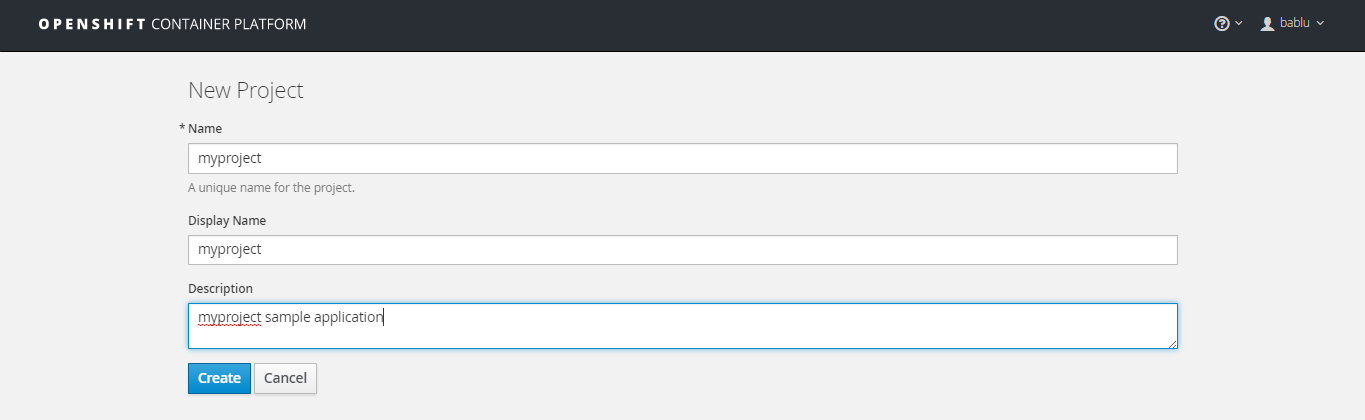
OPENSHIFT

Post Installation Steps

1. Run an application(eureka) using source code (here we are using github as the repository)
2. Login to the openshift and create a new project by clicking on the “New Projet” tab.



1. Give any name of your choice, display name and description of your project and click create.



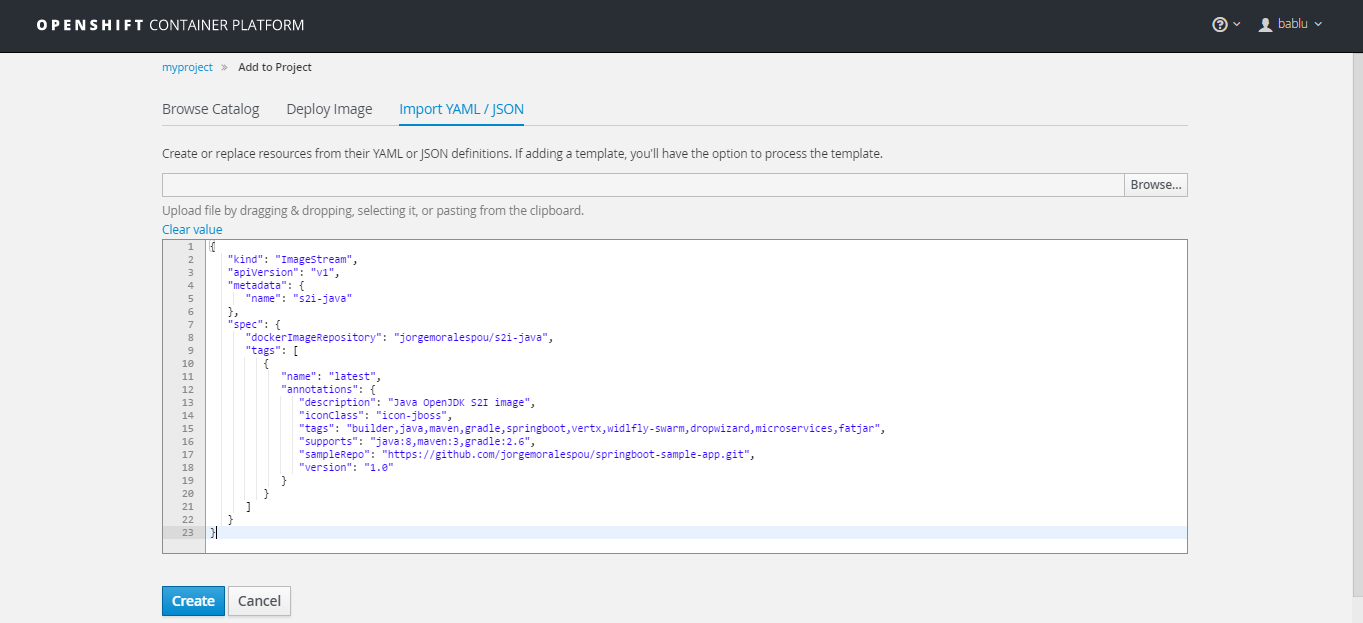
1. We need an imagestream to run our application, in my case I am going to run springboot application so I need specific imagestream to run springboot application.

My version of openshift doesn’t come with any imagestream for running springboot, but openshift allow us to create our own imagestream.

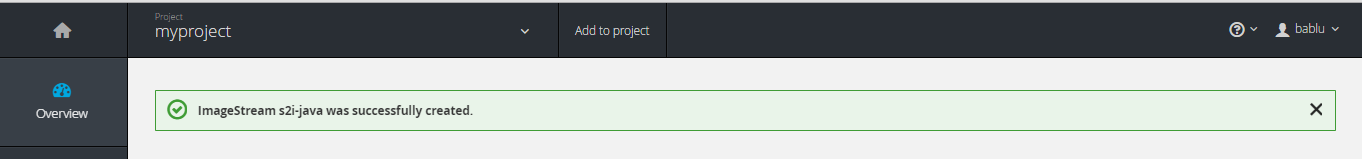
So, create a new imagestream

|  |
| --- |
| {  "kind": "ImageStream",  "apiVersion": "v1",  "metadata": {  "name": "s2i-java"  },  "spec": {  "dockerImageRepository": "jorgemoralespou/s2i-java",  "tags": [  {  "name": "latest",  "annotations": {  "description": "Java OpenJDK S2I image",  "iconClass": "icon-jboss",  "tags": "builder,java,maven,gradle,springboot,vertx,widlfly-swarm,dropwizard,microservices,fatjar",  "supports": "java:8,maven:3,gradle:2.6",  "sampleRepo": "https://github.com/jorgemoralespou/springboot-sample-app.git",  "version": "1.0"  }  }  ]  }  } |

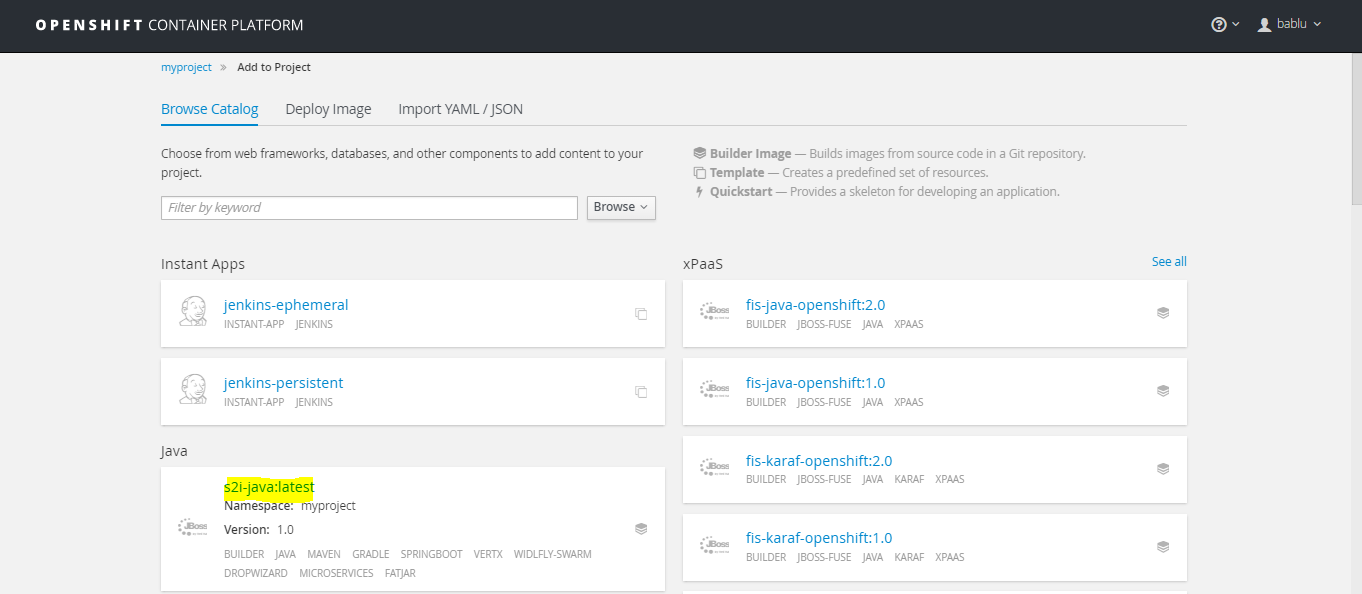
Use this json file to create imagestream. Copy the contents and place it in import YAML/JSON section in openshift console and click create.



You will get a successful message.

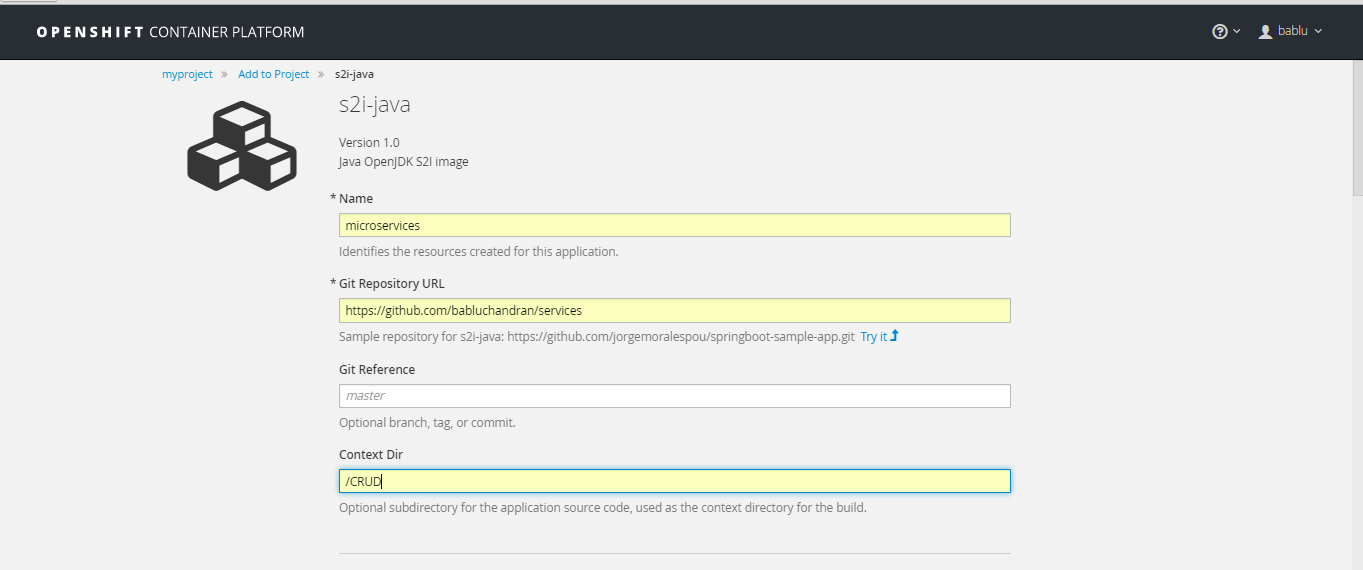


Go and check whether new imagestream is created or not. Click on “Add to project” tab and check for the imagestream.

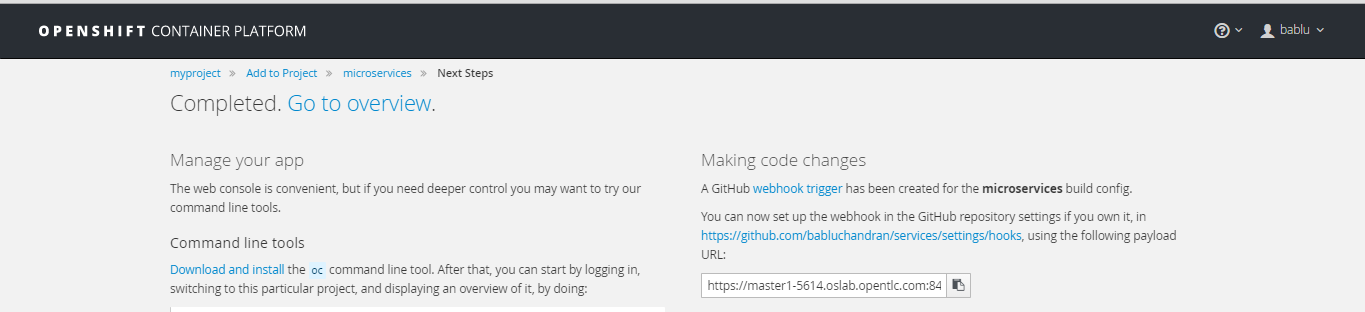


Use this imagestream(s2i-java-latest) to run our springboot application.

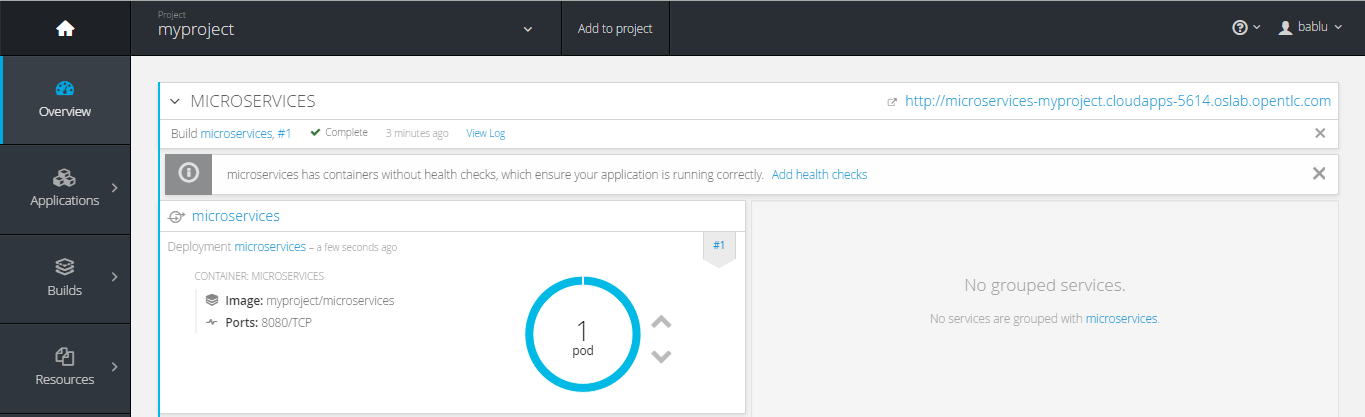
1. Click on the imagestream we created, it will redirect to a new screen. Give the desired application name and the source directory(here Github). Click on the “[Show advanced routing, build, and deployment options](javascript:;)”. If you have multiple directories in your source repository specify individual directories in “**Context Dir**” section(one at a time). Leave other default settings as it is and click create.



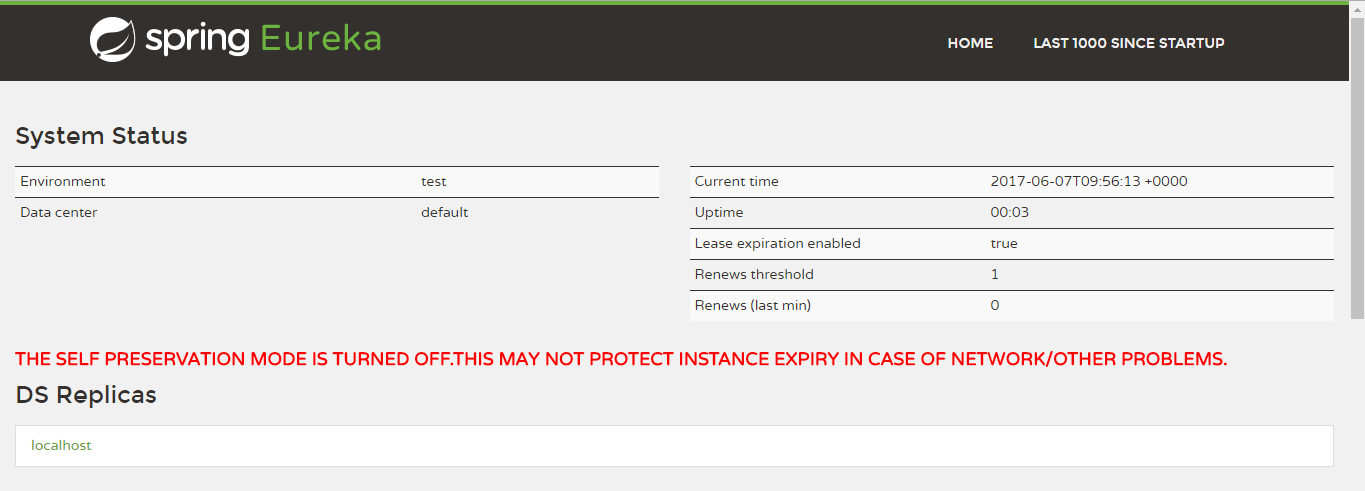
It will return a successful message after creation. Click on “Go to Overview” section



1. You can see your application details in this section. The blue circle indicates application is deployed.



1. Click on the route(http link )to access the application. It will open a new tab and will have access to our application.
2. Here, my eureka server is up and running.



Note:

Deploy microservices application.

1. Follow above steps to create a New Project in openshift.
2. Deploy each services(follow above steps) inside the same project in openshift.
3. Just give your directory name in the “**Context Dir**” section to select a particular services.
4. Running application(CRUD) and Registering CRUD inside eureka.
5. In the properties file of our CRUD application we are using environment variables to get the eureka host and port as shown below. The **${eurekahost}** is a variable and the value for the variable can be assigned during run time of the application.

|  |
| --- |
| spring:  application:  name: zuul-gateway  cloud:  config:  enabled: false  eureka:  instance:  nonSecurePort: ${server.port:9090}  client:  serviceUrl:  defaultZone: ${eurekahost}/eureka/ |

1. Follow the previous steps to create a new-application. In the “[Show advanced routing, build, and deployment options](javascript:;)” add the environment variables(here eurekahost🡪<http://microservices-myproject.cloudapps-5614.oslab.opentlc.com/>) and click create.

This environment variable (eurekahost) is mentioned in the application’s(CRUD) properties file and will identify this env variable to register itself in eureka.

After successful deployment of CRUD, reload the eureka dashboard and we can see that CRUD is registered in eureka.

