

IBM CLOUDPAK WORKSHOP

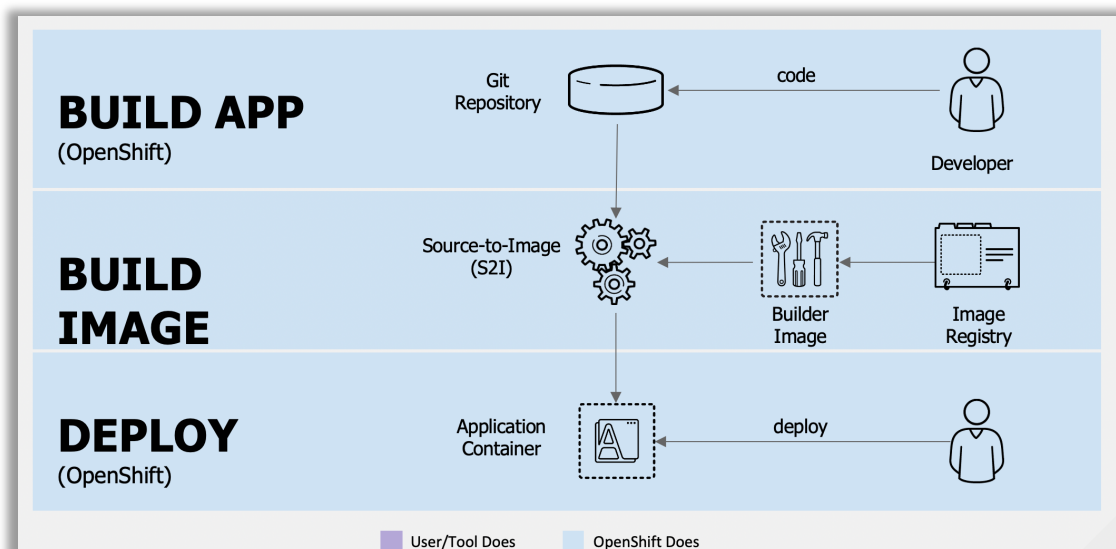
OPENSIFT LABS

Simple Deployment Pipeline from GIT repo

Version: 2020-03-03

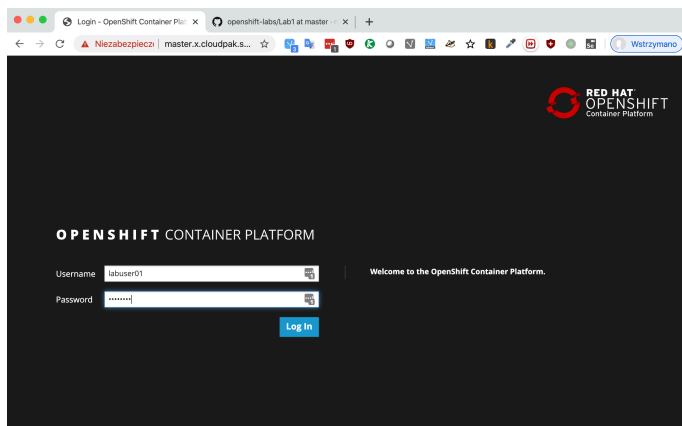
This lab has been created by Maciej Szulc (maciej.szulc@pl.ibm.com) for the IBM CloudPak workshop

In this lab we will get familiar with the Source-To-Image mechanism, that allows to easily deploy the application from Your code repository.



1.LOG IN TO THE CLUSTER AND ACCESS THE CATALOG

1. Go to Your VNC client, open web browser and navigate to our cluster at <https://master.x.cloudpak.site:8443>
2. Login using credentials provided by IBM (Openshift Cluster Account)



3. Navigate to Languages tab shown on “Browse Catalog” screen



2.CREATE APPLICATION FROM PYTHON CODE

Now we will create a configuration that will get the code from github and create all required Kubernetes/Openshift elements.

4. Click on “**Python**” icon



5. Click on “**Python**” to start creation of runtime config. The system will present You with a creator.
6. Click “**Next**” on “Information” step
7. Fill the Application name as **<Your username>-python**, i.e.: labuser01-python.
8. Fill the Git repository as
“<https://github.com/maciej20/IBMCloudPakWorkshop.git>”

9. Click on “Advanced options”

If you have a private Git repository or need to change application defaults, view [advanced options](#).

10. **Our code is not in the root directory** – so we need to fill the context dir.
Find “Context dir” and fill with “**Code/1**”

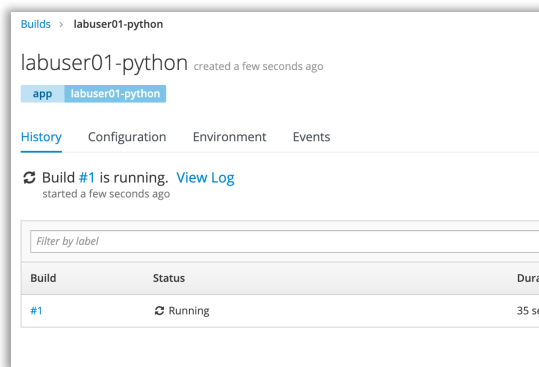
11. Leave all remaining options intact and click “Create” in the bottom of the screen

3. VERIFY THE BUILD PROCESS

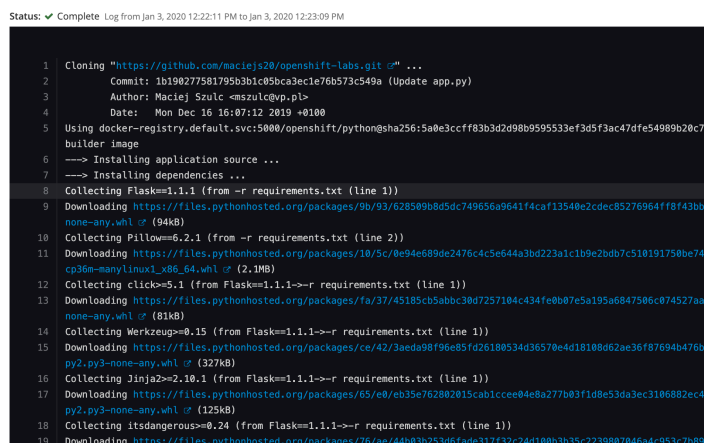
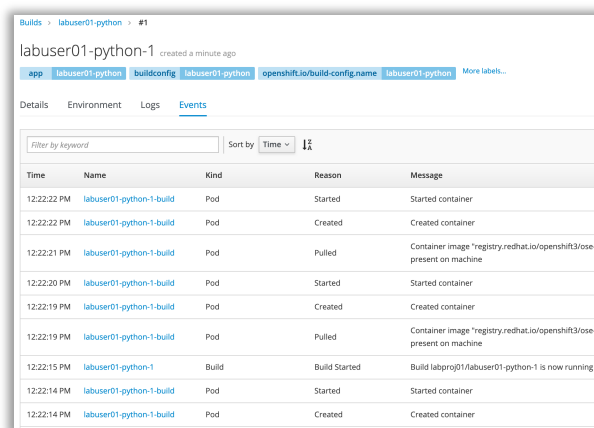
12. Navigate to “Builds” -> “Build” from the menu on the left side of the screen.
Click on “labuser01-python” to see the build details

Name	Last Build	Status	Duration
labuser01-python	#1	Running	19 seconds

13. Click on “#1” to see current build details



14. Navigate to “Logs” and “Events” to see the build process

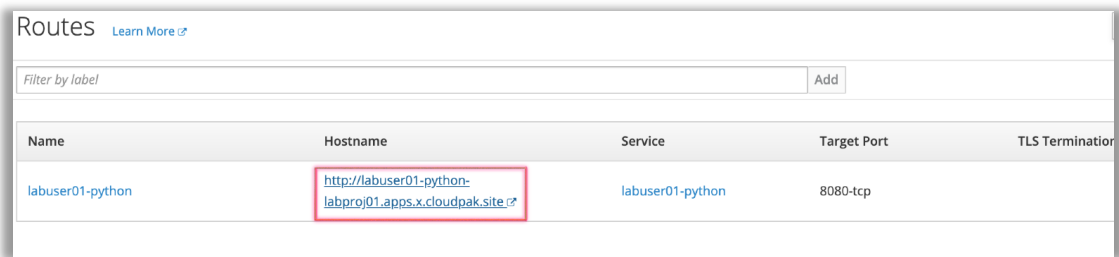


Wait for the app to build.

4. VERIFY IF APP WORKS

15. Use the icon to navigate to “Applications” -> Routes

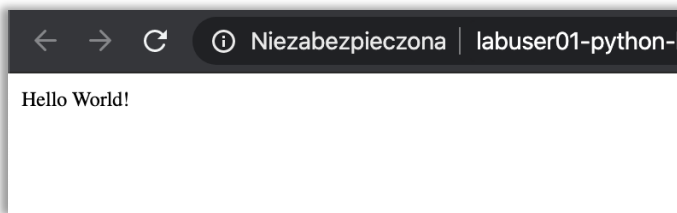
16. Now we have to verify if Your app works.
Click on Your route (will be named <Your_name>-python-<Your project>.apps.x.cloudpak.site)



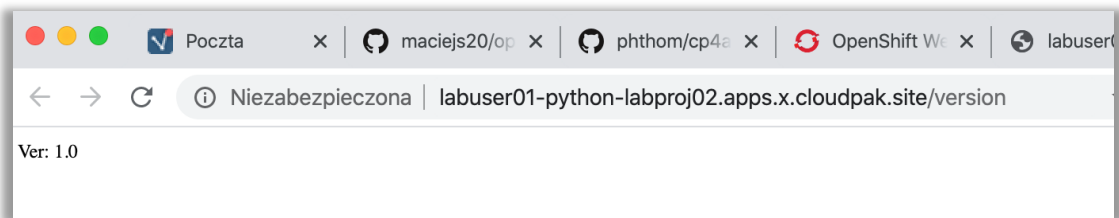
The screenshot shows the 'Routes' page in OpenShift. A table lists the routes. The first row is highlighted, showing the route 'labuser01-python' with the hostname 'http://labuser01-python-labproj01.apps.x.cloudpak.site'. The hostname is highlighted with a red box.

Name	Hostname	Service	Target Port	TLS Termination
labuser01-python	http://labuser01-python-labproj01.apps.x.cloudpak.site	labuser01-python	8080-tcp	

17. Verify if there is a “Hello world” message from app deployed.



18. Check the code version adding /version to the URL bar

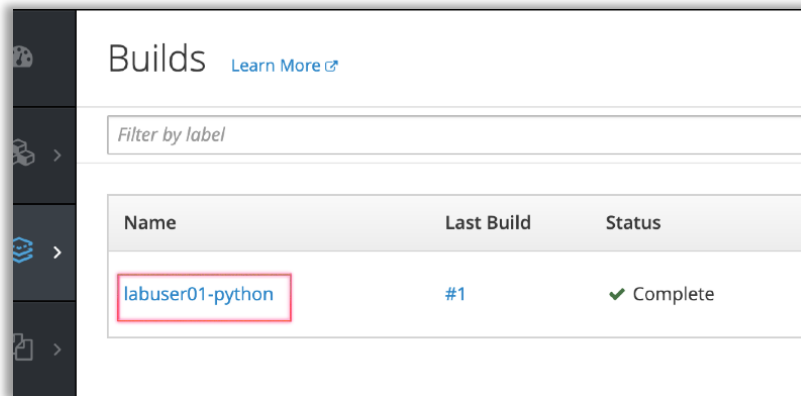


5.CHANGE APPLICATION CODE

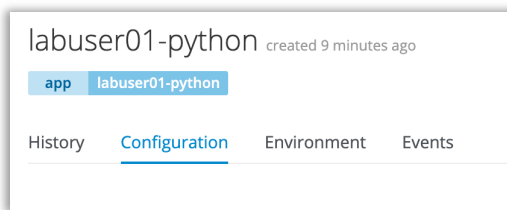
When we are aligning with CI/CD process, it is necessary to use automated build. Openshift S2I can re-build the application with any code change in git repo. For our lab we are using shared code repository that has no access to our cluster, that's why we can't use this mechanism directly.

We will simply switch the context dir to simulate the change instead.

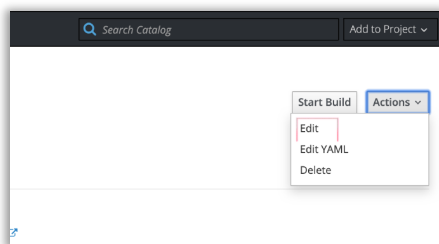
19. Use the  icon to navigate to “Builds” -> “Builds”. Click on Your build (labuserXX-python)



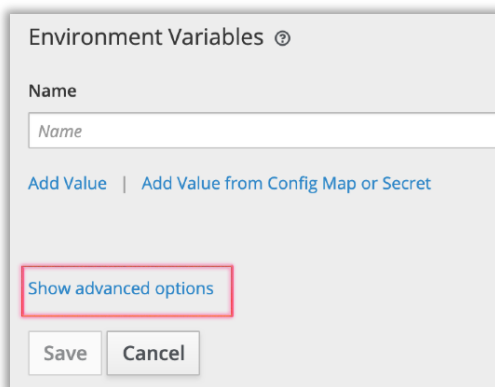
20. Click on **“Configuration”**



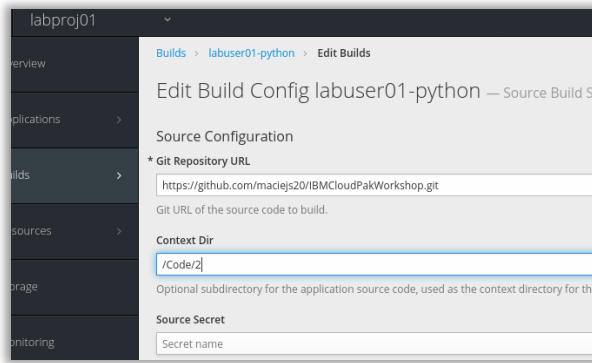
21. From the **“Actions”** dropdown select **“Edit”**



22. Click on **“Show advanced options”** at the bottom of the screen

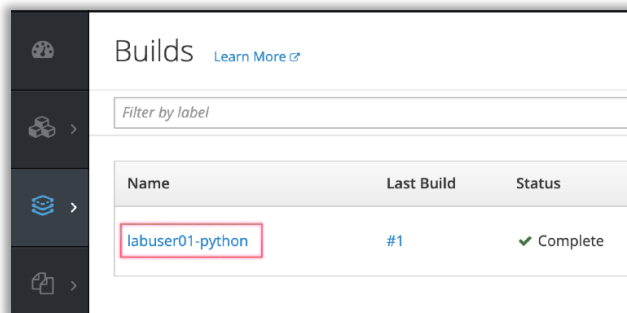


23. Change the **“Context dir”** to **/Code/2**




24. Click on **“Save”** on the bottom of the screen.

25. Use the  icon to navigate to **“Builds”** -> **“Builds”**. Click on Your build.



26. In a production environment git calls the Openshift to automatically rebuild the application on each code change. As we can’t use such mechanism in our lab environment, we have to re-build the app manually.

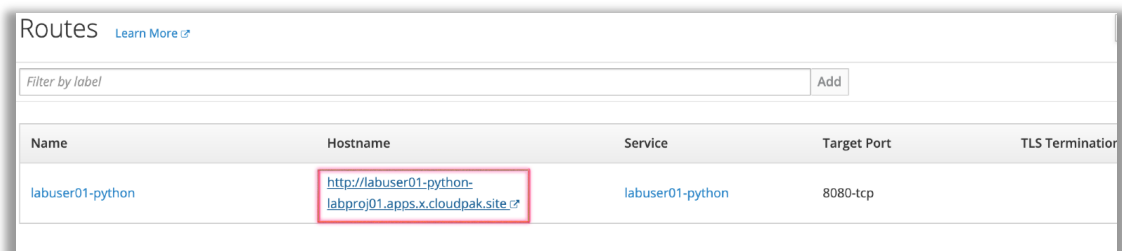
Click on the **Start Build**  button and wait for the build to complete.

6. VERIFY IF NEW APP WORKS

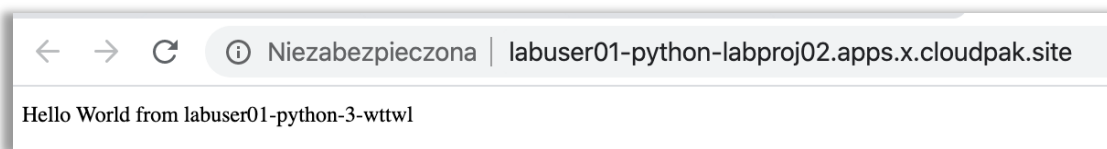
27. Use the  icon to navigate to **“Applications”** -> **Routes**

28. Now we have to verify if Your app works.

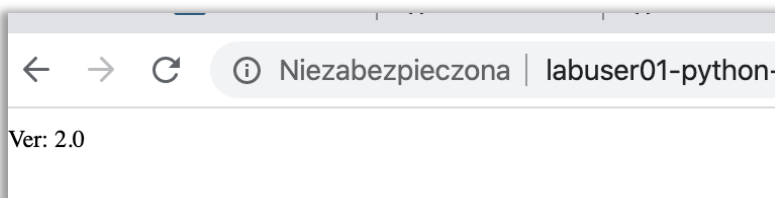
Click on Your route (will be named <Your_name>-python-<Your project>.apps.x.cloudpak.site)




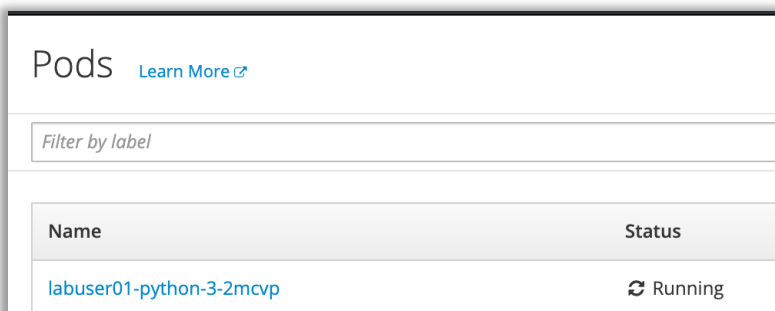
29. Verify if there is a “Hello world from <hostname>” message from app deployed. This version of the lab displays the hostname of the pod it is running on.



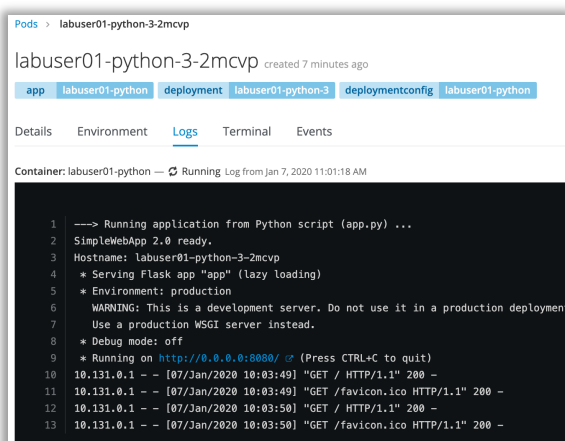
30. Check the code version adding /version to the URL bar



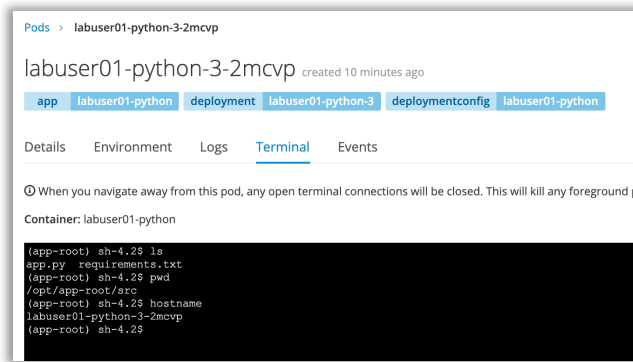
31. Openshift allows to interact with Your pods easily. Use the  icon to navigate to “Applications” -> “Pods”. Click on one of the running pods for Your app.



32. Go to “Logs” tab to see the logs from Your running application.



33. Click on “Terminal” to connect to command line session inside the application pod.
Enter “ls -l” to see the directory content in the pod.
Enter “hostname” to see the pod hostname

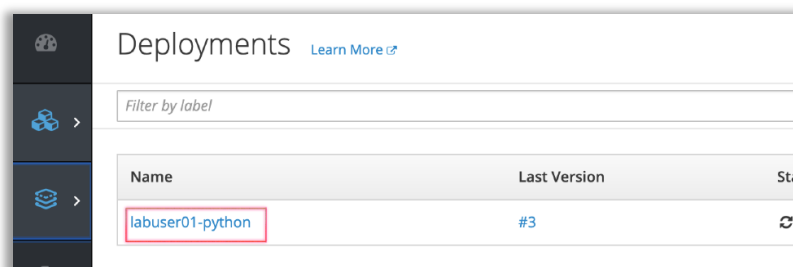


7. USE AUTOSCALLER

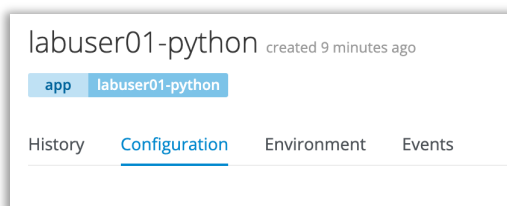
The build mechanisms in the Openshift allows to configure many things that should be configured manually in Kubernetes.

In the previous lab we have been using the GUI to manually increase and decrease the number of pods for the application. Now we will add an autoscaler to scale the number of pods to the resource consumption.

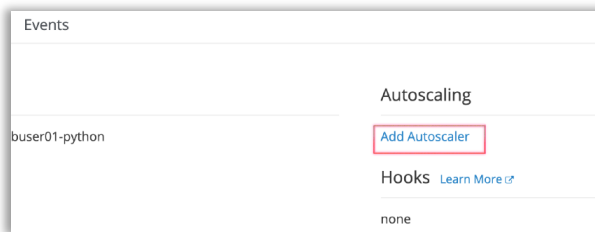
34. Use the  icon to navigate to “Applications” -> **Deployments**. Click on Your deployment



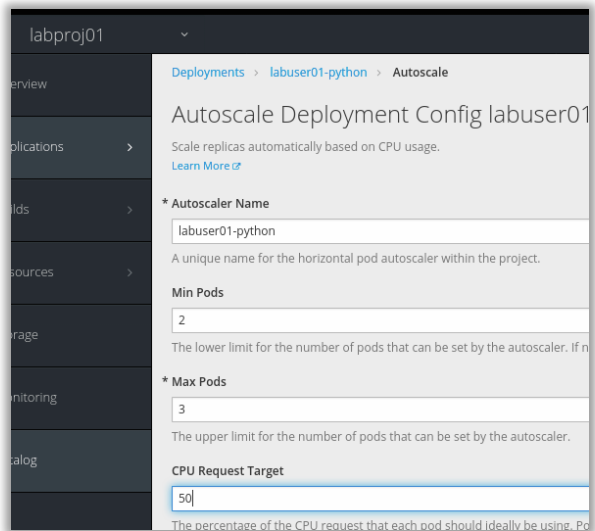
35. Click on “Configuration” tab



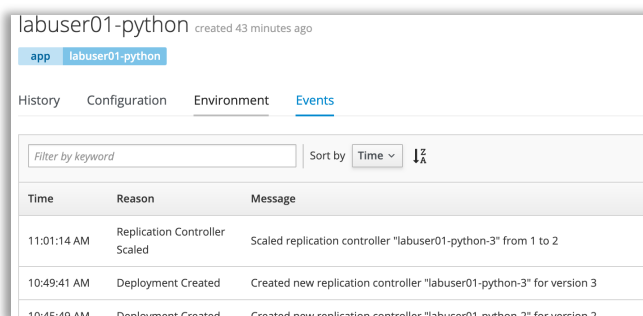
36. Click on “Add autoscaler”



37. Set the “Min Pods” to 2, “Max Pods” to 3 and “CPU Request Target” to 50.
Leave other options intact.



38. Click on “Save”
39. Click on “Events” to see autoscaler in work. It may take a minute for
autoscaler to add the app instance.



40. Use the  icon to navigate to “Applications” -> “Pods” to see the number
of pods


There should be 2 pods running.

Pods [Learn More](#)

Filter by label

Name	Status	Containers Ready
labuser01-python-3-2mcvp	Running	1/1
labuser01-python-3-wttwl	Running	1/1
labuser01-python-2-build	Completed	0/1
labuser01-python-1-build	Completed	0/1

8. VERIFY IF SCALED APP WORKS

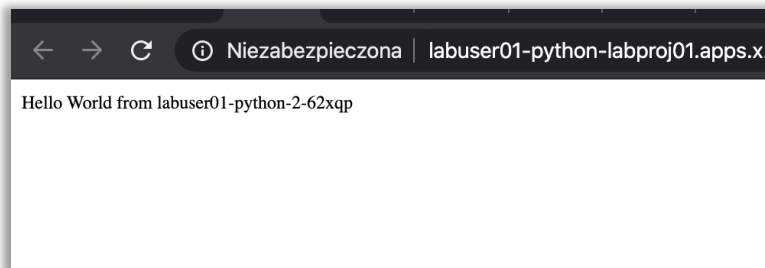
41. Use the  icon to navigate to “Applications” -> Routes
42. Click on Your route (will be named <Your_name>-python-<Your project>.apps.x.cloudpak.site)

Routes [Learn More](#)

Filter by label

Name	Hostname	Service	Target Port	TLS Termination
labuser01-python	http://labuser01-python-labproj01.apps.x.cloudpak.site	labuser01-python	8080-tcp	

43. Verify if there is a “Hello world” message from app deployed.




We have two pods running but usually You will see only single hostname in the response. Default route handling policy sticks the session (together with pod) to a cookie, so each time You are presented with data from the same pod.

You may change this behavior with the lab below **or just clear the cookies and refresh the page.**

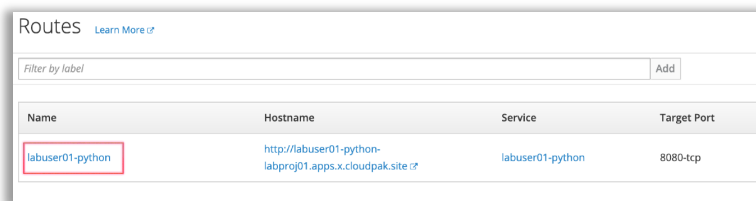
9.CHANGE ROUTE ANNOTATIONS (OPTIONAL)

Openshift routes are using Annotations to configure specific settings. All the details are described in the manual. We will use two annotations to change the route behavior:

- haproxy.router.openshift.io/balance – to set the policy
- haproxy.router.openshift.io/disable_cookies – to disable 'per-cookie' pod selection

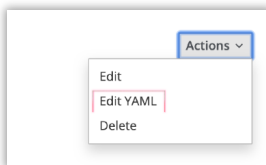
44. Use the  icon to navigate to “Applications” -> Routes

45. Click on the “labuserxx-python” route config to open route config



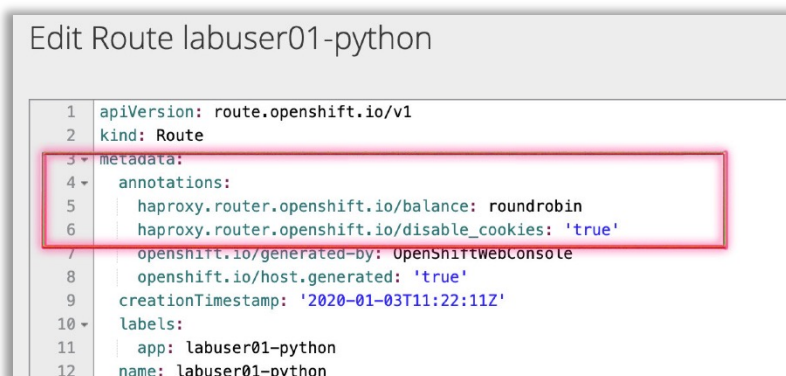
Name	Hostname	Service	Target Port
labuser01-python	http://labuser01-python-labproj01.apps.x.cloudpak.site	labuser01-python	8080-tcp

46. Click on “Actions” -> “Edit YAML”



47. Add following settings to “annotation” setting. Double check that - **Yaml depends on the number of spaces** – so it has to be like on the picture below!

```
haproxy.router.openshift.io/balance: roundrobin
haproxy.router.openshift.io/disable_cookies: 'true'
```



48. Wait for 2-3 minutes, open app and verify if the host name changes when refreshing the app page.

It may require several page refreshes to see the host changing.

