IBM Node-RED IBM Cloud template 2020. Getting source code

LAB-0-02 - getting source code

Вступ

Документація написана з використанням загадльно-прийнятого формату markdown (файли типу *.md). Короткий довідник по markdown знаходиться по лінку markdown help.

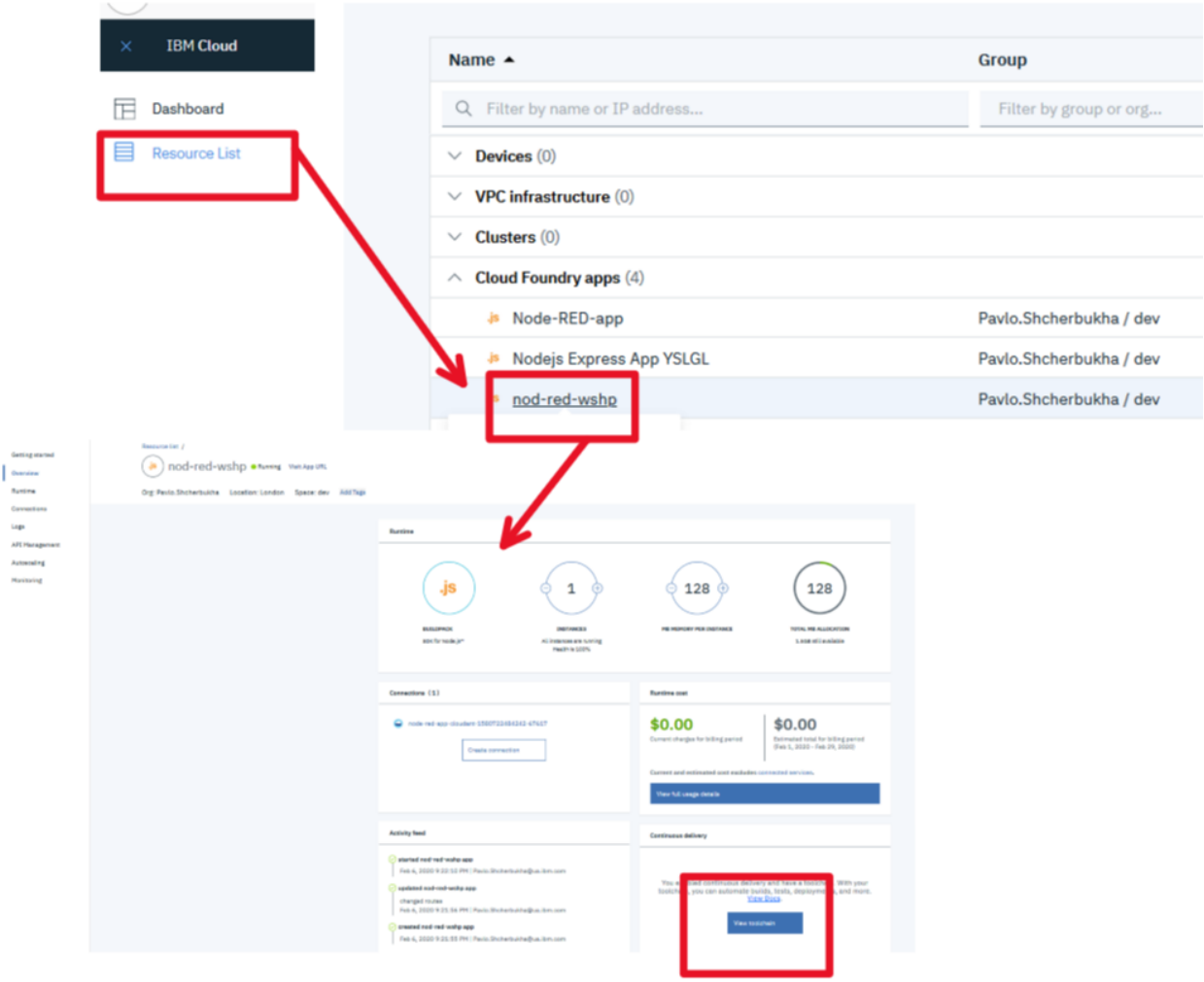
Лабораторні роботи розраховані на роботу з OS windows-10

В IBM Cloud уже збережено програмний код. Для його отримання потрібно:

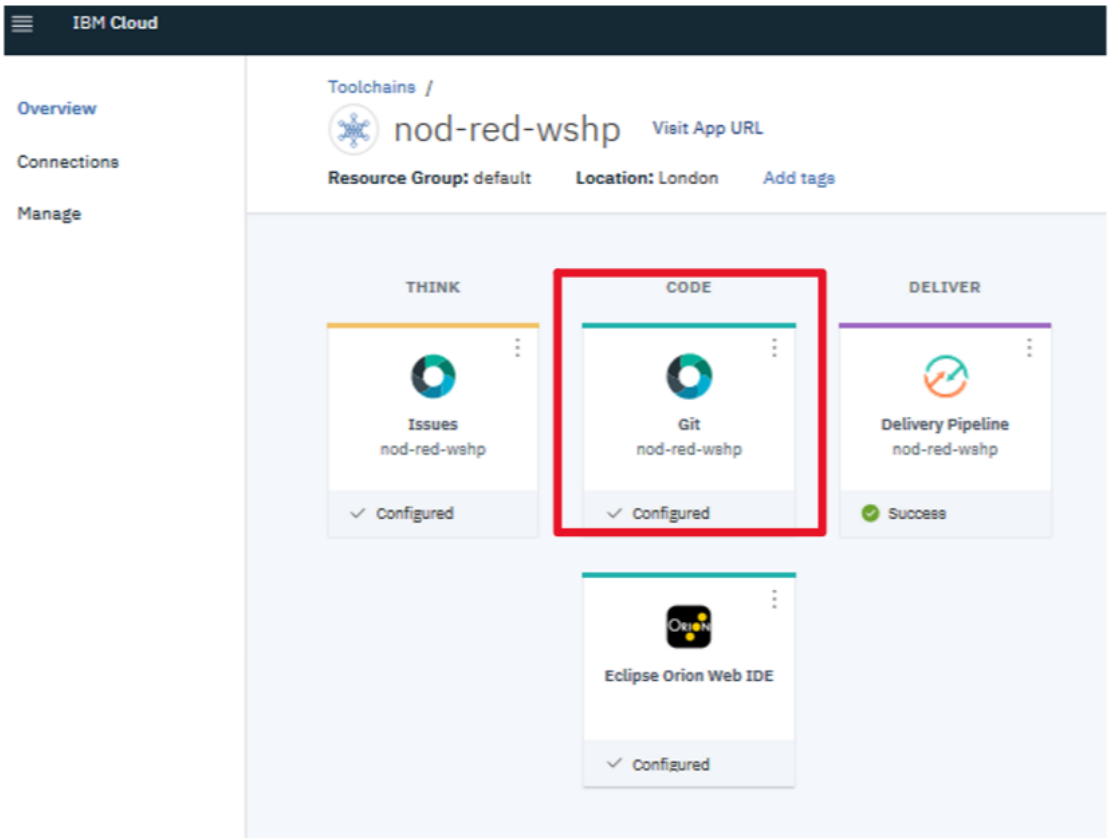
- знайти URL git-репозиторію
- згенеруати необхідні credentials
- клонувати git-репозиторій на робочу станцію
- відкрити висхідний код в Visual Code Studio

Робота з IBM Cloud git. знайти URL git-репозиторію

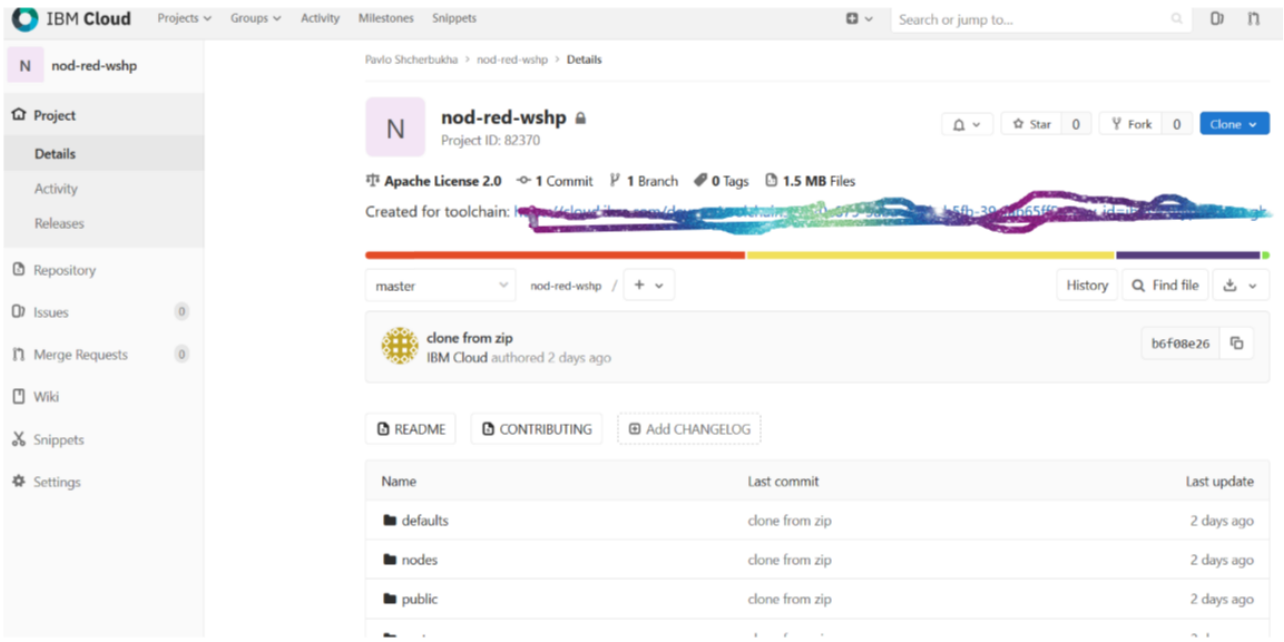
- Спершу в IBM Cloud знаходимо наш додаток те переходимо в його deployment toolchain. Детально кроки показані на малюнку:


IBM Cloud
Dashboard
Resource List
Name
Group
Filter by name or IP address...
Filter by group or org...
Devices (0)
VPC infrastructure (0)
Clusters (0)
Cloud Foundry apps (4)
Node-RED-app
Pavlo.Shcherbukha / dev
Nodejs Express App YSLGL
Pavlo.Shcherbukha / dev
nod-red-wshp
Pavlo.Shcherbukha / dev
nod-red-wshp
$0.00
$0.00

- В результаті виконання цих кроків попадаємо в deployment toolcain



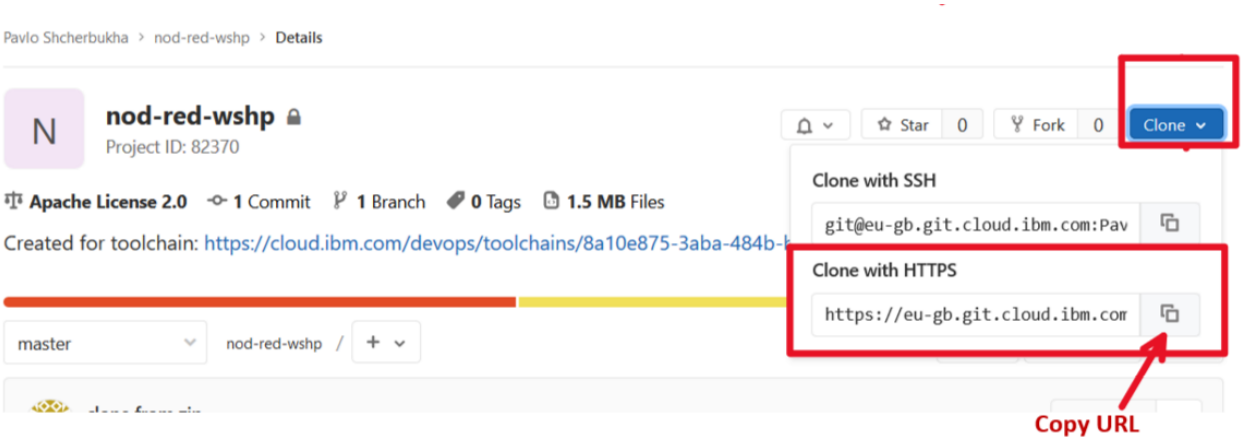
- Шляхом кліку на іконці Git (відмічена червоним), попадаємо в git-repo IBM-Cloud



На цьому скріншоті ми можемо бачити, що це звичайний git - репозиторій, що нічим не відрізняється від github чи github.

Для роботи з remote репозиторієм потрібно:

- отримати http url репозиторію для колнування. На цьому скріншоті показано як знайти URL для клонування

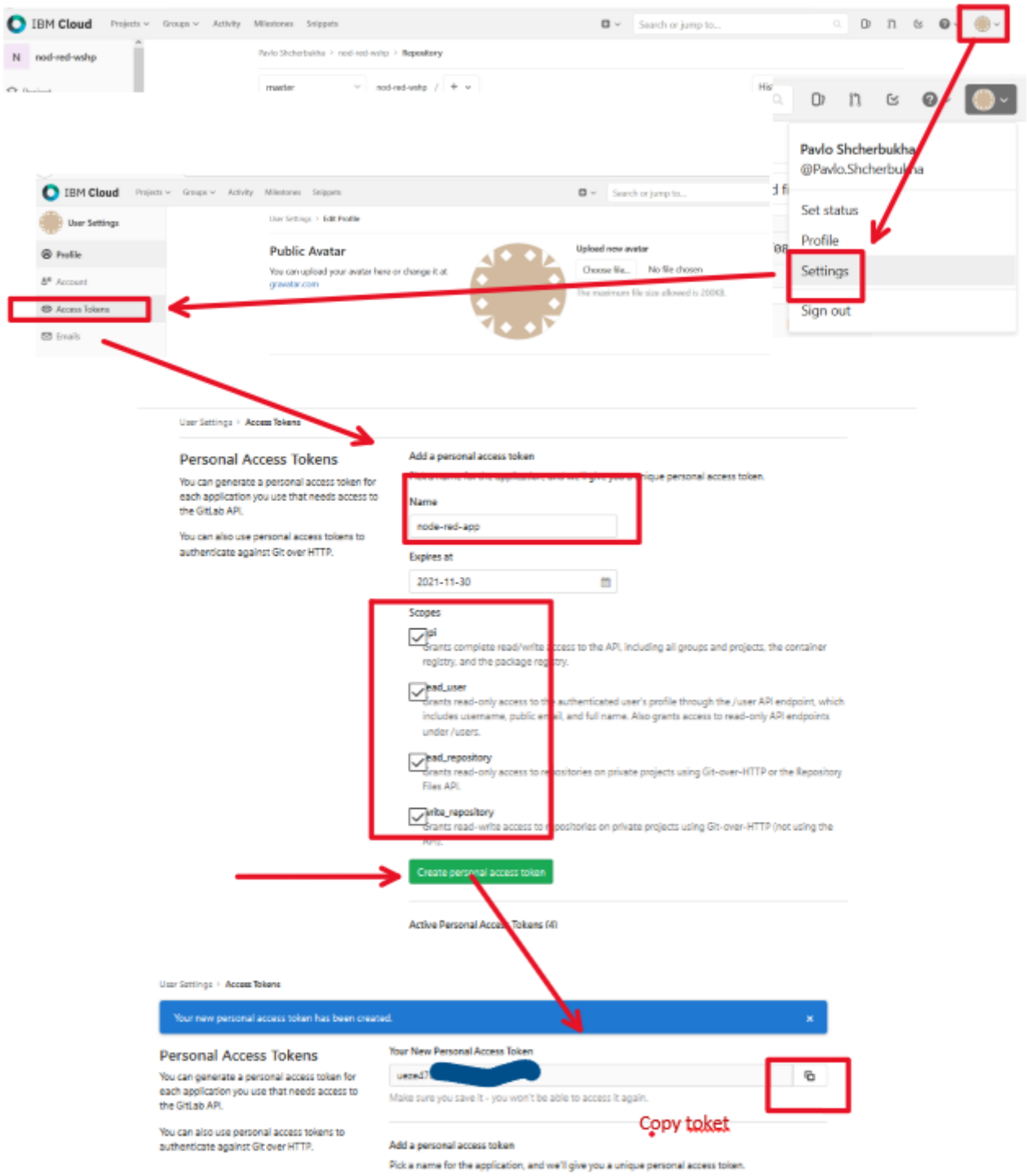


Але потрібно мати на увазі, що цей URL буде модифікований credantials. Як отримати credentials - читаємо в наступних пунктах

Робота з IBM Cloud git. Згенеруати необхідні credentials

Credentials для автоматичного доступу до репозиторію складаються з git username та git token

Процес генерації токена показаний на скріншоті



A git username показаній на малюнку у правому верхньому куті біля знака "@" **Pavlo.Shcherbukha**

Згенерований токен потрібно зберегти, в наступному розділі він буде підставлятися в URL.

Робота з IBM Cloud git. Клонування репозиторію на локальну станцію

Для клонування порібно створити каталого, на приклад Lab0-02-app Клонування виконується командою:

```
git clone [url-repo] -b master [path] , де
```

- [url-repo] - url репозиторію має таку структуру https://git-username:git-token@url
- -b master - вказує на branch, з якого потрібо зробити клон, в нашому випадку master
- [path] - повний шлях до репозиторію, але через слеш /

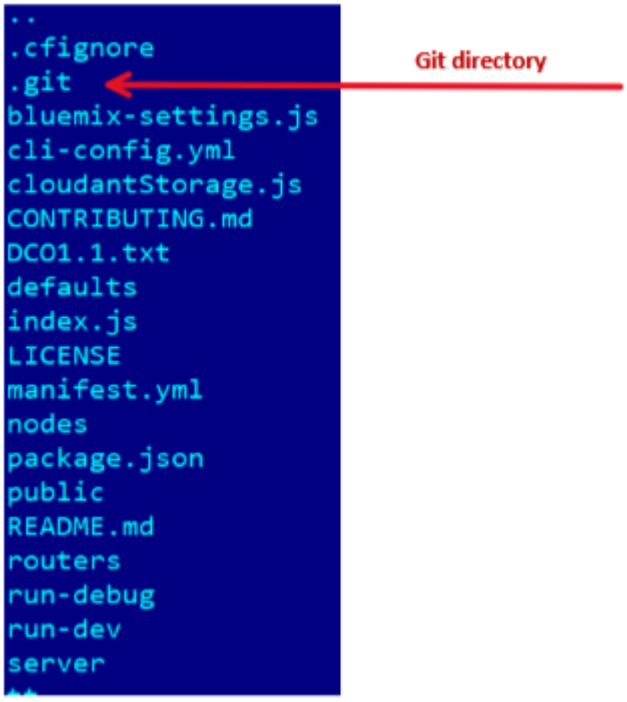
Приклад:

```
git username: Pavlo.Pavlo
git token: xxxxxxxxxxxxxxxxxx
git url repo: https://eu-gb.git.cloud.ibm.com/Pavlo.Pavlo.Pavlo/nod-red-wshp.git
```

clone url with credentials:

```
https://Pavlo.Pavlo:xxxxxxxxxxxxxxxxxx@eu-gb.git.cloud.ibm.com/Pavlo.Pavlo/nod-red-wshp.git
```

В результаті клонування отримаємо щось схоже на це:



Робота з IBM Cloud git. Додаткові настройки Git

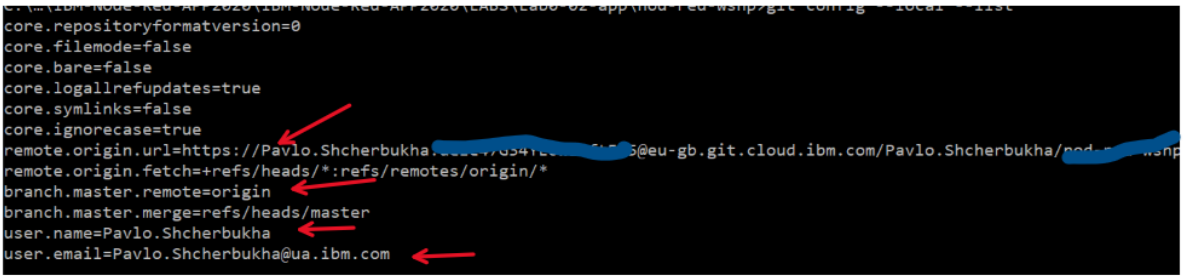
Для повноти настройки бажано прописати в локальному git ваш логин та e-mail, ті ж самі, що і в локальному репозиторії

```
    git config user.name "Pavlo.Pavlo"
    git config user.email "Pavlo.Pavlo@xx.yy.com"
```

Результат конфігурації можна попачити, винонавши команду:

```
git config --local --list
```

Команда поверне, щось схоже на це:



Робота з IBM Cloud git. Змінити код та відпавити його знову в IBM Cloud

На цьому етапі зробимо невеликі зміни в отриманому коді. Наприклад в файлі readme.md додамо рядок з текстом:

```
TEST UPLOAD
```

потім, відправимо код в репозиторій. Це викличе перебудову всього хмарного додатку. Це буде приклад тестового deployment. Тож,

- Міняємо readme.md
- Додаємо файл під контроль git

```
git add README.md
  or
git add *.*
```

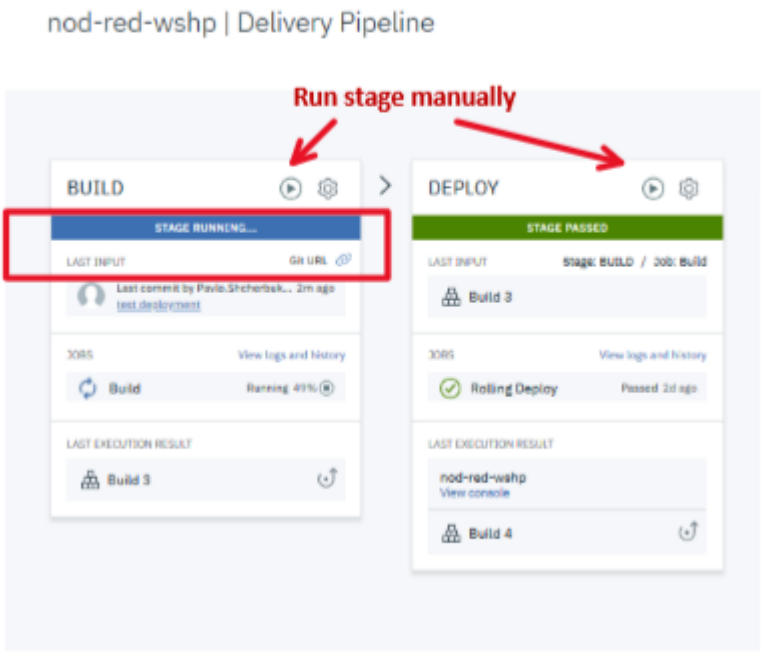
- Виконуємо commit в локальний репозиторій командою:

```
git commit -m "test deployment"
```

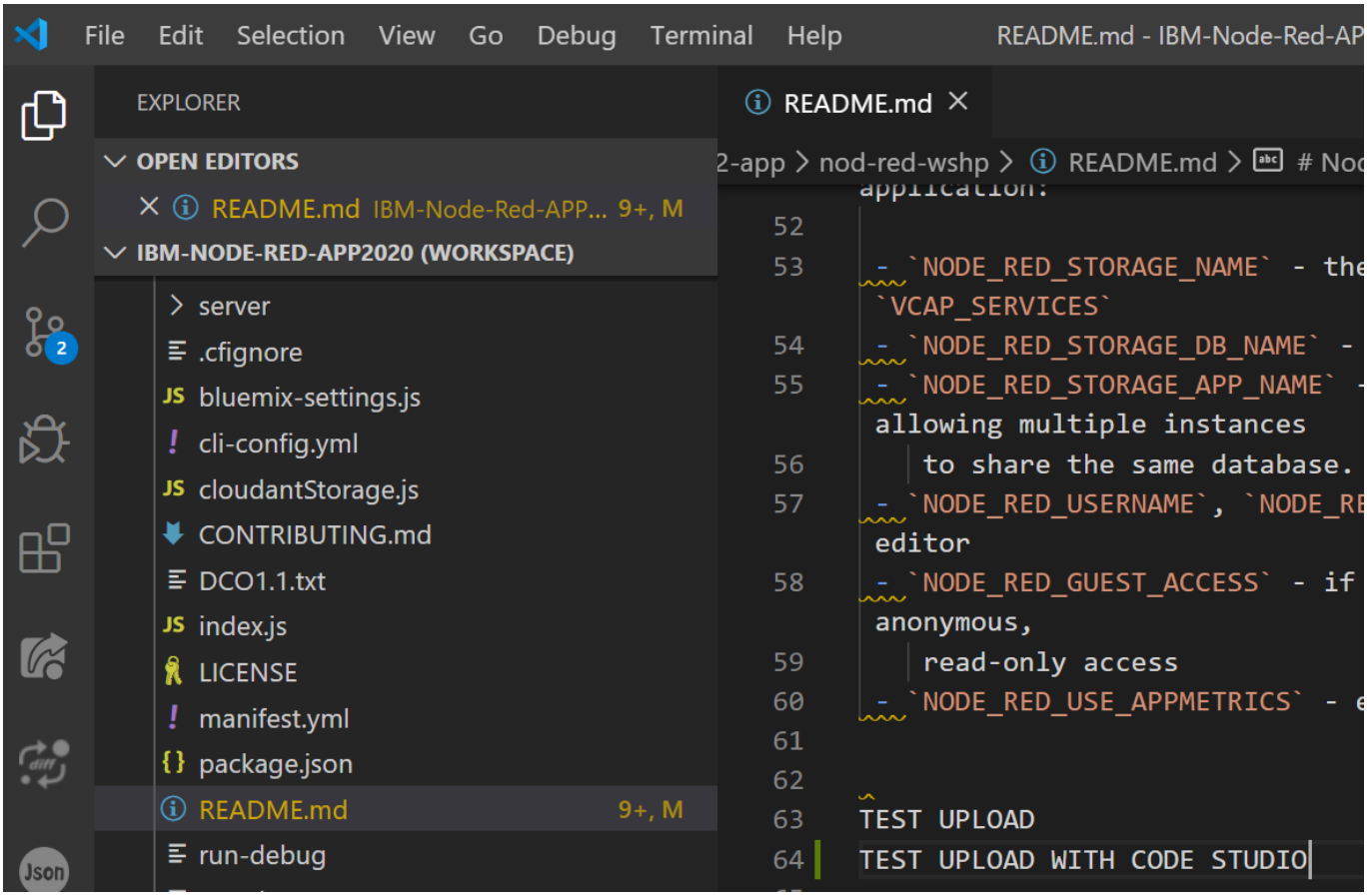
- Відправляємо зміни в хмарний репозиторій командою:

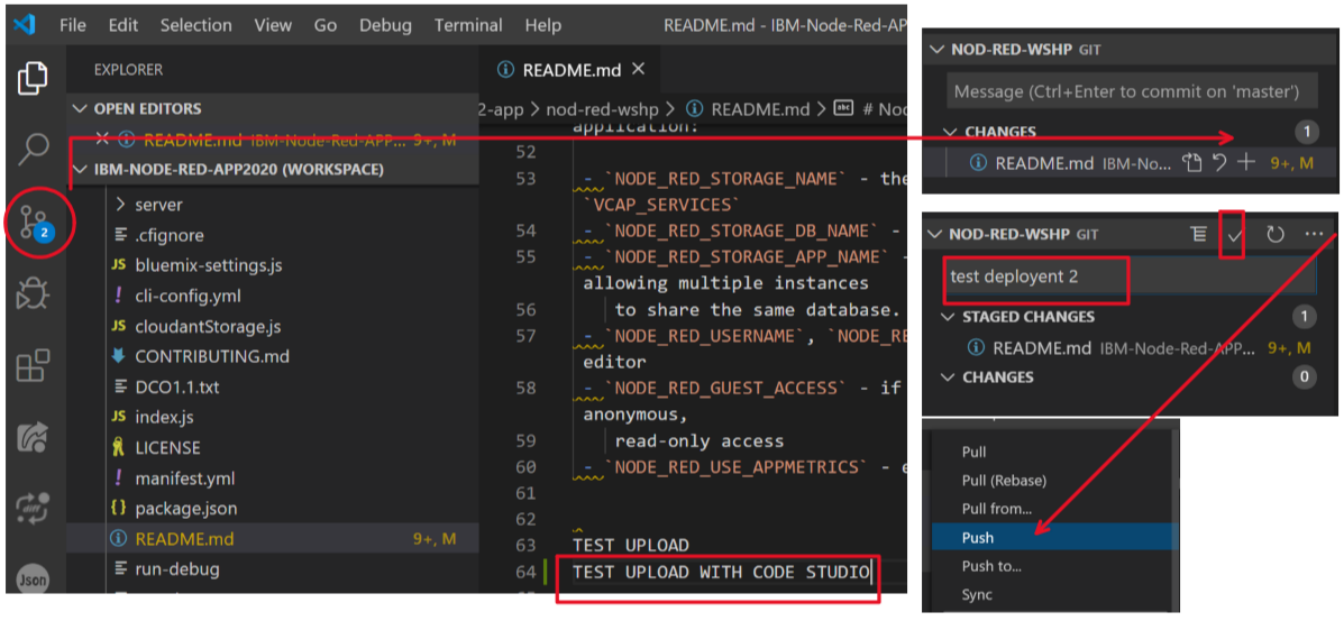
```
git push
```

- Захрдимо в Delivery pipeline Тут видно як зразу запуствся deployment нашого додатку.



Тепер відкриемо README.md з допомогою Code studio то отримаємо такий же результат, але командами code studio





Deployent з допомогою IBM Cloud CLI

Є ще один шлях, що дозволяє заделоїти наше app прямо з локальної станції прямо в IBM Cloud. Це, використання команд ibmCloud cf (IBM CloudFoundry). Depoyment відбувається поза toolchain. Для Deployment з допомогою команд:

- ibmCloud
- Working with the Cloud Foundry CLI

Для deployment основним конфігураційним файлом при цьому являється manifest.yml. При deployemnt потрібно посилатися на нього. В наг=шому каталозі він присутній, тому, що цими ж командами виконуєиться deployment в toolchain в блоці Deployment. Таким чином для deployment виконується така послідовніст команд:

```
    ## login    key --sso for ibm employee
    ibmCloud login --sso

    ## select cloudfoundry
    ibmcloud target --cf

    ## deployment
    ibmCloud cf push -f C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-
app\nod-red-wshp\manifest.yml
```

Ну, а це deployment log. Тобто все ок.

```
PS C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-red-wshp> ibmCloud
cf push -f C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-red-
wshp\manifest.yml
```

```
Invoking 'cf push -f C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-
red-wshp\manifest.yml'...

Pushing from manifest to org Pavlo.Shcherbukha / space dev as
Pavlo.Shcherbukha@ua.ibm.com...
Using manifest file C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-
red-wshp\manifest.yml
Deprecation warning: Use of 'buildpack' attribute in manifest is deprecated in
favor of 'buildpacks'. Please see http://docs.cloudfoundry.org/devguide/deploy-
apps/manifest.html#deprecated for alternatives and other app manifest
deprecations. This feature will be removed in the future.



Deprecation warning: Route component attributes 'domain', 'domains', 'host',
'hosts' and 'no-hostname' are deprecated. Found: domain, host.
Please see http://docs.cloudfoundry.org/devguide/deploy-
apps/manifest.html#deprecated for the currently supported syntax and other app
manifest deprecations. This feature will be removed in the future.


Using manifest file C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-
red-wshp\manifest.yml

Creating app nodredwshp in org Pavlo.Shcherbukha / space dev as
Pavlo.Shcherbukha@ua.ibm.com...
OK

Using route nod-red-wshp.eu-gb.mybluemix.net
Binding nod-red-wshp.eu-gb.mybluemix.net to nodredwshp...
OK

Uploading nodredwshp...
Uploading app files from: C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-
app\nod-red-wshp
Uploading 38.6K, 39 files
Done uploading
OK
Binding service node-red-app-cloudant-1580722484242-67617 to app nodredwshp in org
Pavlo.Shcherbukha / space dev as Pavlo.Shcherbukha@ua.ibm.com...
OK

Starting app nodredwshp in org Pavlo.Pavlo / space dev as Pavlo.Pavlo@xx.yy.com...
Downloading sdk-for-nodejs...
Downloaded sdk-for-nodejs
Cell b9a2e90b-32cf-4fd8-8ae4-29123528c50f creating container for instance
4b564114-702d-454f-8afc-f4f40ae3477d
Cell b9a2e90b-32cf-4fd8-8ae4-29123528c50f successfully created container for
instance 4b564114-702d-454f-8afc-f4f40ae3477d
Downloading app package...
Downloaded app package (37.2K)
-----> IBM SDK for Node.js Buildpack v4.1-20191119-1309
       Based on Cloud Foundry Node.js Buildpack 1.7.3
```

```
-----> Installing binaries
       engines.node (package.json): 12.x
       engines.npm (package.json): unspecified (use default)
       Attempting to install: 12.13.0
-----> Installing node 12.13.0
       Copy
[/tmp/buildpacks/4bbe598ac8a30281a9532d35fdaaf98d/dependencies/5c1414e90396c08e1f8
97e8855a34a90/node-12.13.0-linux-x64-cflinuxfs3-55d69507.tgz]
       Using default npm version: 6.12.0
-----> Installing yarn 1.19.1
       Copy
[/tmp/buildpacks/4bbe598ac8a30281a9532d35fdaaf98d/dependencies/3c6e4143e3d81d24d93
8953b6c5437e0/yarn-1.19.1-any-stack-34293da6.tar.gz]
       Installed yarn 1.19.1
-----> Creating runtime environment
       PRO TIP: It is recommended to vendor the application's Node.js dependencies
       Visit http://docs.cloudfoundry.org/buildpacks/node/index.html#vendoring
       NODE_ENV=production
       NODE_HOME=/tmp/contents058899009/deps/0/node
       NODE_MODULES_CACHE=true
       NODE_VERBOSE=false
       NPM_CONFIG_LOGLEVEL=error
       NPM_CONFIG_PRODUCTION=true
-----> Building dependencies
       Installing node modules (package.json)
> ibm_db@2.6.3 install /tmp/app/node_modules/ibm_db
> node installer/driverInstall.js
platform =  linux , arch =  x64 , node.js version =  v12.13.0
make version = GNU Make 4.1
Downloading DB2 ODBC CLI Driver from
https://public.dhe.ibm.com/ibmdl/export/pub/software/data/db2/drivers/odbc_cli/lin
uxx64_odbc_cli.tar.gz...
0.04% | 8192 bytes downloaded out of 21032500 bytes.
0.08% | 16384 bytes downloaded out of 21032500 bytes.
0.12% | 24576 bytes downloaded out of 21032500 bytes.
0.16% | 32768 bytes downloaded out of 21032500 bytes.
0.19% | 40960 bytes downloaded out of 21032500 bytes.
0.23% | 49152 bytes downloaded out of 21032500 bytes.
0.27% | 57344 bytes downloaded out of 21032500 bytes.
99.87% | 21004288 bytes downloaded out of 21032500 bytes.
99.90% | 21012480 bytes downloaded out of 21032500 bytes.
99.94% | 21020672 bytes downloaded out of 21032500 bytes.
99.98% | 21028864 bytes downloaded out of 21032500 bytes.
100.00% | 21032500 bytes downloaded out of 21032500 bytes.
************************************
You are downloading a package which includes the Node.js module for IBM
DB2/Informix.  The module is licensed under the Apache License 2.0. The package
also includes IBM ODBC and CLI Driver from IBM, which is automatically downloaded
as the node module is installed on your system/device. The license agreement to
the IBM ODBC and CLI Driver is available in undefined   Check for additional
dependencies, which may come with their own license agreement(s). Your use of the
components of the package and dependencies constitutes your acceptance of their
respective license agreements. If you do not accept the terms of any license
agreement(s), then delete the relevant component(s) from your device.
```

```
**************************************
Downloading and extraction of DB2 ODBC CLI Driver completed successfully ...
make: Entering directory '/tmp/app/node_modules/ibm_db/build'
  CXX(target) Release/obj.target/odbc_bindings/src/odbc.o
  CXX(target) Release/obj.target/odbc_bindings/src/odbc_connection.o
  CXX(target) Release/obj.target/odbc_bindings/src/odbc_statement.o
  CXX(target) Release/obj.target/odbc_bindings/src/odbc_result.o
  SOLINK_MODULE(target) Release/obj.target/odbc_bindings.node
  COPY Release/odbc_bindings.node
make: Leaving directory '/tmp/app/node_modules/ibm_db/build'
> websocket@1.0.31 install /tmp/app/node_modules/websocket
> (node-gyp rebuild 2> builderror.log) || (exit 0)
make: Entering directory '/tmp/app/node_modules/websocket/build'
  CXX(target) Release/obj.target/bufferutil/src/bufferutil.o
  SOLINK_MODULE(target) Release/obj.target/bufferutil.node
  COPY Release/bufferutil.node
  CXX(target) Release/obj.target/validation/src/validation.o
  SOLINK_MODULE(target) Release/obj.target/validation.node
  COPY Release/validation.node
make: Leaving directory '/tmp/app/node_modules/websocket/build'
> bcrypt@3.0.6 install /tmp/app/node_modules/@node-red/editor-
api/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build
make: Entering directory '/tmp/app/node_modules/@node-red/editor-
api/node_modules/bcrypt/build'
  CXX(target) Release/obj.target/bcrypt_lib/src/blowfish.o
  CXX(target) Release/obj.target/bcrypt_lib/src/bcrypt.o
  CXX(target) Release/obj.target/bcrypt_lib/src/bcrypt_node.o
  SOLINK_MODULE(target) Release/obj.target/bcrypt_lib.node
  COPY Release/bcrypt_lib.node
  COPY /tmp/app/node_modules/@node-red/editor-
api/node_modules/bcrypt/lib/binding/bcrypt_lib.node
  TOUCH Release/obj.target/action_after_build.stamp
make: Leaving directory '/tmp/app/node_modules/@node-red/editor-
api/node_modules/bcrypt/build'
> bcrypt@3.0.6 install /tmp/app/node_modules/node-red/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build
make: Entering directory '/tmp/app/node_modules/node-
red/node_modules/bcrypt/build'
  CXX(target) Release/obj.target/bcrypt_lib/src/blowfish.o
  CXX(target) Release/obj.target/bcrypt_lib/src/bcrypt.o
  CXX(target) Release/obj.target/bcrypt_lib/src/bcrypt_node.o
  SOLINK_MODULE(target) Release/obj.target/bcrypt_lib.node
  COPY Release/bcrypt_lib.node
  COPY /tmp/app/node_modules/node-
red/node_modules/bcrypt/lib/binding/bcrypt_lib.node
  TOUCH Release/obj.target/action_after_build.stamp
make: Leaving directory '/tmp/app/node_modules/node-red/node_modules/bcrypt/build'
> bcrypt@3.0.8 install /tmp/app/node_modules/bcrypt
> node-pre-gyp install --fallback-to-build
[bcrypt] Success: "/tmp/app/node_modules/bcrypt/lib/binding/bcrypt_lib.node" is
installed via remote
> jsonpath@1.0.2 postinstall /tmp/app/node_modules/jsonpath
> node lib/aesprim.js > generated/aesprim-browser.js
```

```
added 606 packages from 572 contributors and audited 2608 packages in 52.656s
found 13 vulnerabilities (3 low, 8 moderate, 2 high)
  run `npm audit fix` to fix them, or `npm audit` for details
       **WARNING** Unmet dependencies don't fail npm install but may cause runtime
issues
       See: https://github.com/npm/npm/issues/7494
       Contrast Security no credentials found. Will not write environment files.
Exit status 0
Uploading droplet, build artifacts cache...
Uploading droplet...
Uploading build artifacts cache...
Uploaded build artifacts cache (18.6M)
Uploaded droplet (59.8M)
Uploading complete
Cell b9a2e90b-32cf-4fd8-8ae4-29123528c50f stopping instance 4b564114-702d-454f-
8afc-f4f40ae3477d
Cell b9a2e90b-32cf-4fd8-8ae4-29123528c50f destroying container for instance
4b564114-702d-454f-8afc-f4f40ae3477d
Cell b9a2e90b-32cf-4fd8-8ae4-29123528c50f successfully destroyed container for
instance 4b564114-702d-454f-8afc-f4f40ae3477d

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started


OK

App nodredwshp was started using this command `npm start`

Showing health and status for app nodredwshp in org Pavlo.Pavlo / space dev as
Pavlo.Pavlo@xx.yy.com...
OK

requested state: started
instances: 1/1
usage: 128M x 1 instances
urls: nod-red-wshp.eu-gb.mybluemix.net
last uploaded: Sat Feb 8 16:02:11 UTC 2020
stack: cflinuxfs3
buildpack: sdk-for-nodejs

     state     since                    cpu    memory          disk
details
#0   running   2020-02-08 06:04:13 PM   0.0%   68.6M of 128M   245.7M of 1G
PS C:\PSHDEV\PSH-WorkShops\IBM-Node-Red-APP2020\Lab0-02-app\nod-red-wshp>
```

Додати нові вузли та задеплоїти в IBM Cloud через GIT репозитрій

Є традиційна задача для Node-Red - додавання нових вузлів та бібліотек. Через меню можна додати - аде воно буде працювати до наступної попудови контейнера. Тому, це не варіант

В принципі, можна зайти в git-repo та внести зміни в package.json. Але, особисто мені такий підхід не подобається і вважаю його не процесним - тобто теж не варіант. У нас є копія на локальній станції можемо зробити традиційний npm install, а потім відправити зміни у git. Далі спацює toolchain і нові бібліотеки будуть доступні для розробки flow.

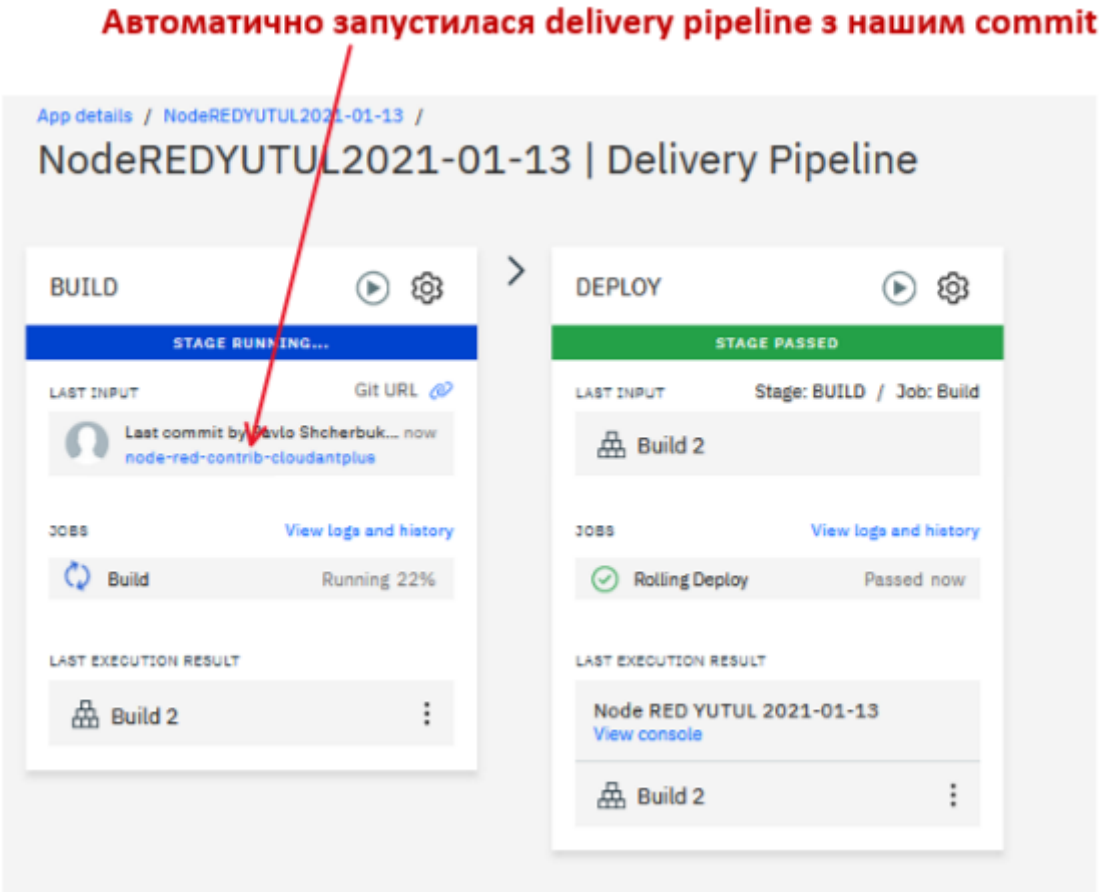
Спробуємо додати бібліотеку по робеті з БД cloudant: node-red-contrib-cloudantplus. по цьому лінку вказано, що установки через npm потрібно виконати команду:

```
npm install node-red-contrib-cloudantplus
```

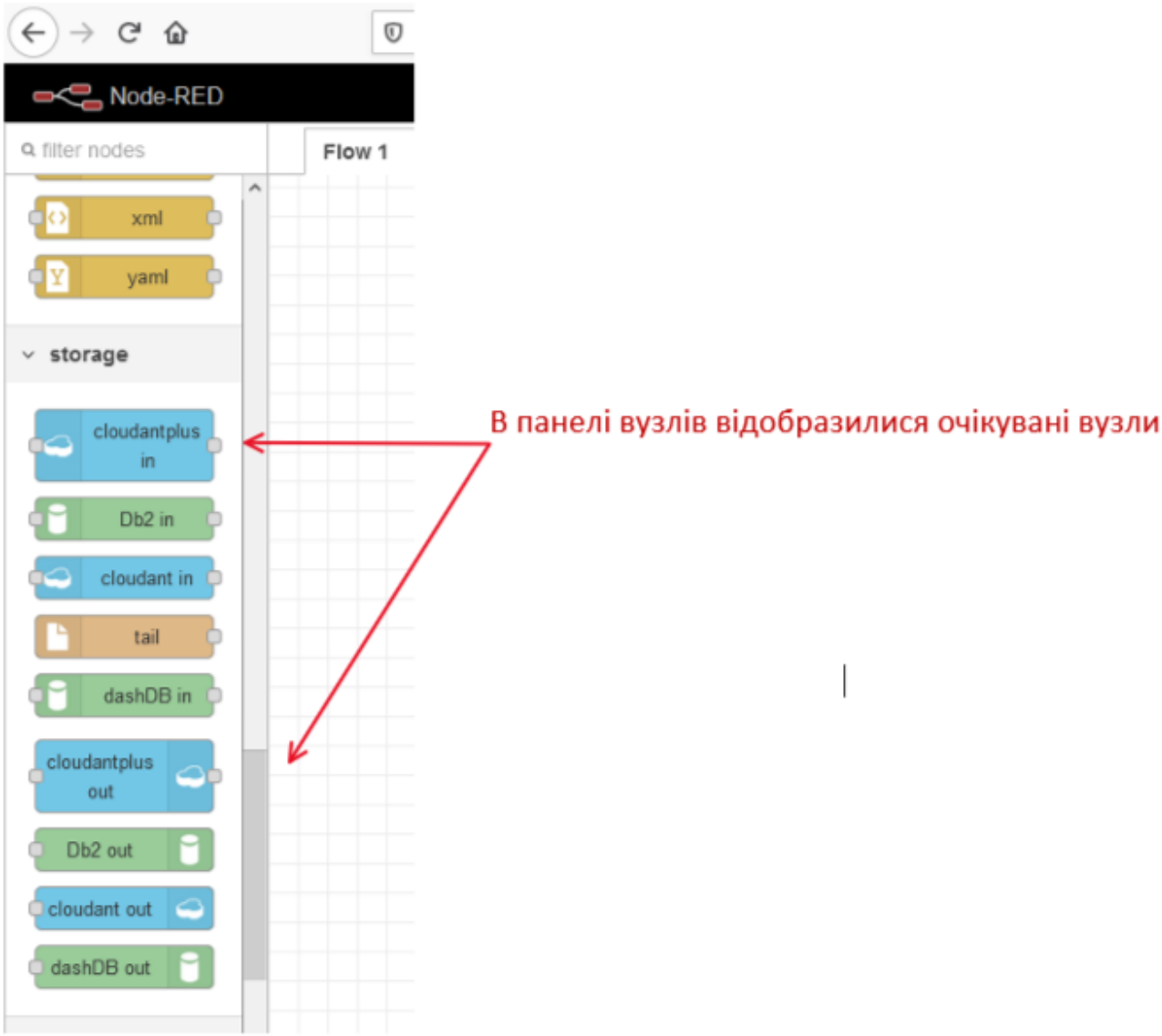
Та пересвідчуємося, що бібліотека з'явилася в package.json

виконуємо git commit та git push - запускається перебудова нашого контейнеру.

На малюнку видно, що автоматично запустився процесс Delivery pipeline

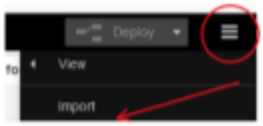


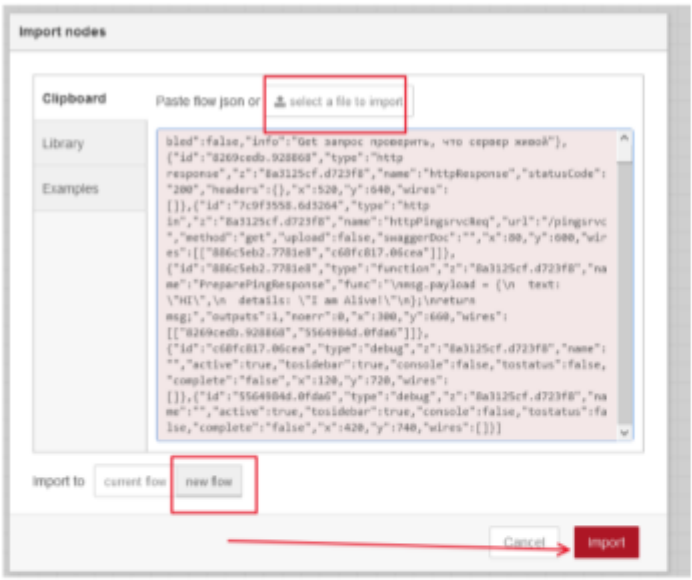
і після запуску нового контейнеру бачимЮ що очікувані вузли з`явилися

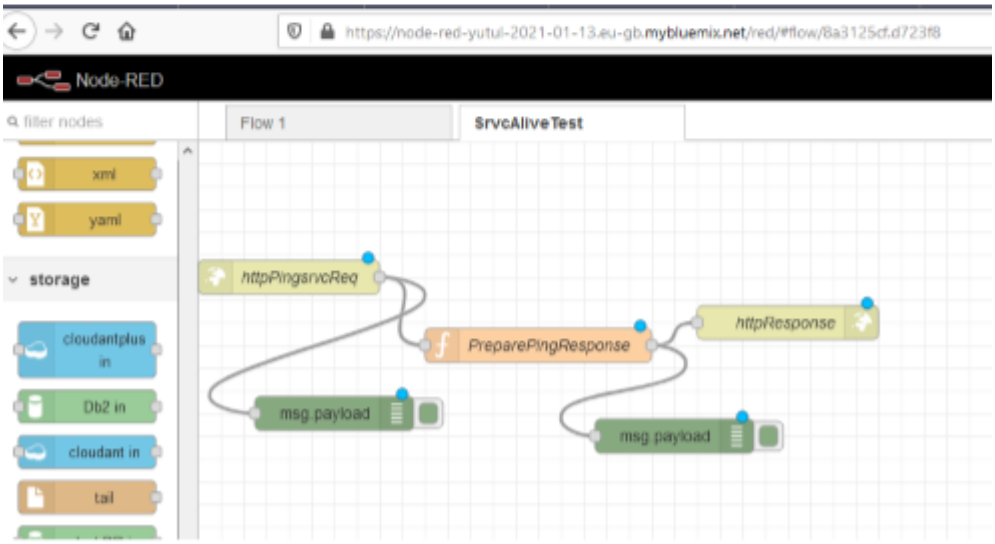


Запуск простого Flow, що денонструє доступність сервісу.

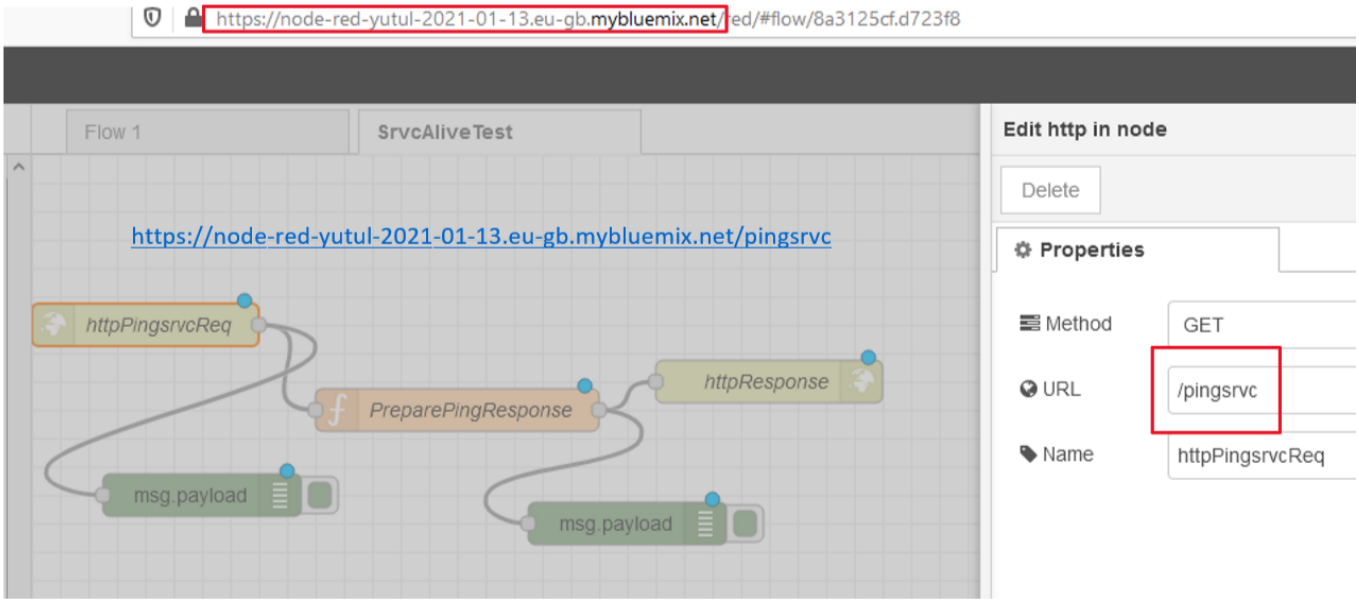
Для цього потрібно імпортувати flow з файлу: **srvcAliveTest_msgflow.json**



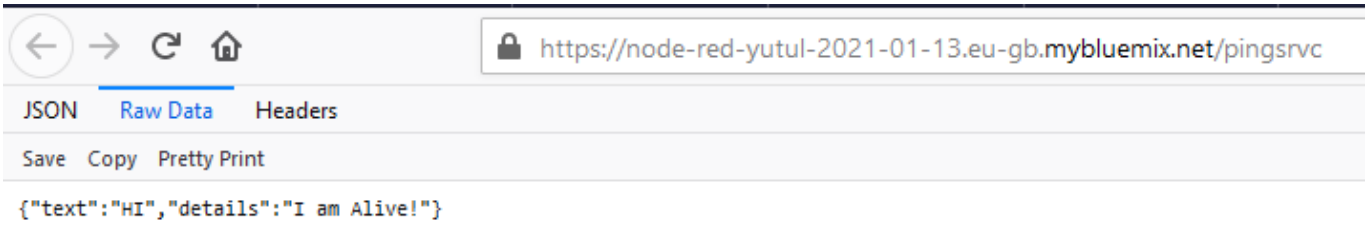




URL flow складається з імені домену, що закінчується на "mybluemix.net/" та суфіксу вузла "httpPingsrvcReq"



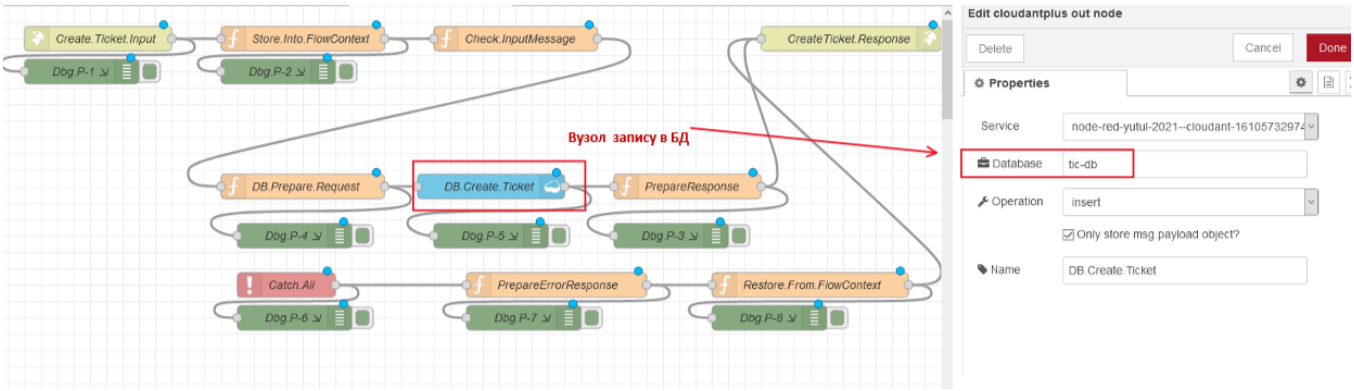
Вбиваємо отриманий URL в адресний рядок браузера і отримаємо відповідь



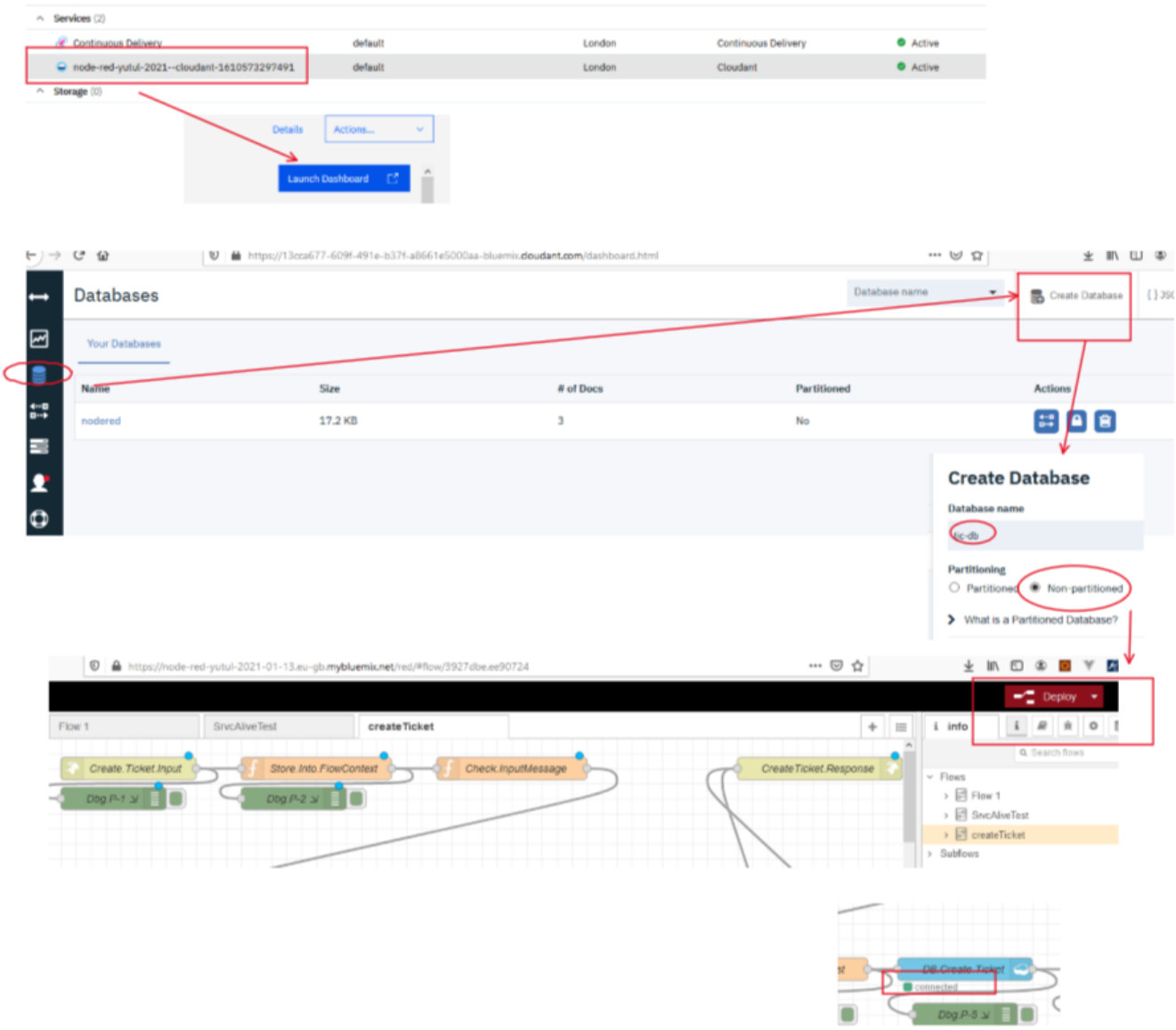
що означає, що app доступне.

Запуск простого Flow, що записує дані в БД Cloudant.

Зпустимо flow що працює з БД cloudant, створюючи умовні тікети. Імортуємо flow з файлу: createTicket_msgflow.json На млюнку, показано вузол запису в БД та назку БД, що потрібно створити в Cloudant



Тепер створимо базу даних, та задеплоїмо flow. Якщо операція пройшла успішно, то біля вузла БД з'явиться значок "conected".



Тепер любим доступним методом робимо http POST

- Http headers

```
content-type: application/json
accept: application/json
```

- Body

```
{"tikUserName": "userK",
"tikUserAddress": "ADDRESSS",
"tikUserMessage": "I have a problem with my home internet",
"tikType": "ISSUE"}
```

І отримаємо відповідь

```
{"ticketnum":"ddd3aad8534b65ddc31c50c5d890ddbf","ticketdt":"01.01.2019
23:12:15","ticketmsg":"Ваша заявка прийнята та зараэстрована"}
```

А в БД залишиться заявка, як на малюнку.