



## Cloud Developer Certification Preparation

### Exercise 5.5:

**Using the Build & Deploy option to manage continuous integration and continuous delivery**

## **Exercise 5.5: Prerequisites**

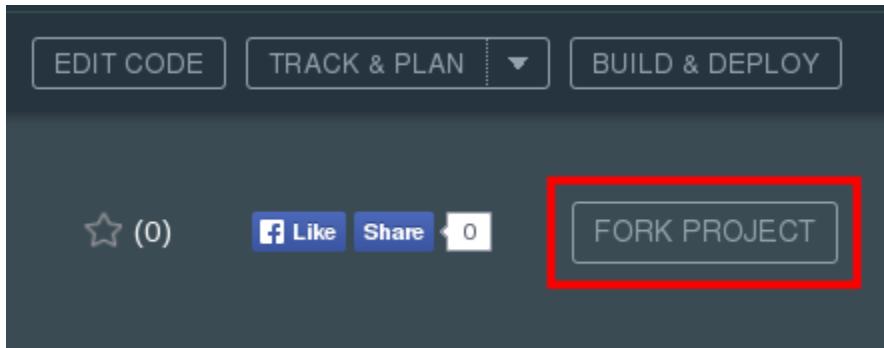
Sign up for a 30-day free trial [IBM Bluemix account](#) if you don't already have one. Also, create a DevOps account: <https://hub.jazz.net/>. You should use the same credentials for both the Bluemix and DevOps accounts.

You also need the following software:

- A web browser supported by Bluemix:
  - Chrome: the latest version for your operating system
  - Firefox: the latest version for your operating system and ESR 31 or ESR 38
  - Internet Explorer: version 10 or 11
  - Safari: the latest version for the Mac

## Exercise 5.5.1: Forking a project and changing some HTML code

1. Log in to IBM DevOps Services and fork the project located at <https://hub.jazz.net/project/ecosysdevcnc/cdc-lab-5/overview> into a new project.



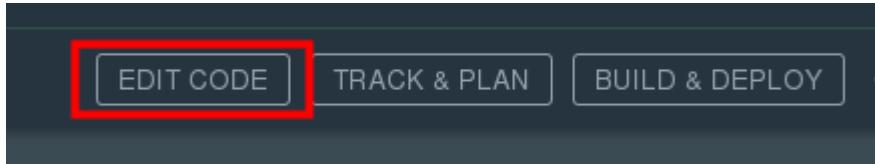
2. Give the new project a unique name, select all the check boxes, and choose an appropriate Bluemix runtime configuration. Click **CREATE**.

The screenshot shows a modal dialog titled 'Fork Project'. It contains the following fields and options:

- Name your project:  (This field is highlighted with a red box.)
- URL: <https://hub.jazz.net/project/holocron/cdc-lab-5e>
- Private project (Invited team members only)
- Restrict membership (IBM only)  
You can restrict this project's membership because your email address ends with ibm.com. If this project is for **IBM confidential** business, you must select this option and agree to certain conditions. [Learn more](#)
- I accept the terms and conditions
- Add features for Scrum development i
- Make this a Bluemix Project i (This checkbox has a red arrow pointing to it.)
- Select a Bluemix space to bill your services to:
  - Region: IBM Bluemix US South
  - Organization: vmorris@us.ibm.com
  - Space: dev
- These selections can be changed later in the options for your Project Settings.
- Buttons: CANCEL and CREATE (The 'CREATE' button is highlighted with a red box.)

3. After you see the message about successfully creating your project, click **EDIT CODE**.

## Exercise 5.5: Configuring a continuous delivery pipeline



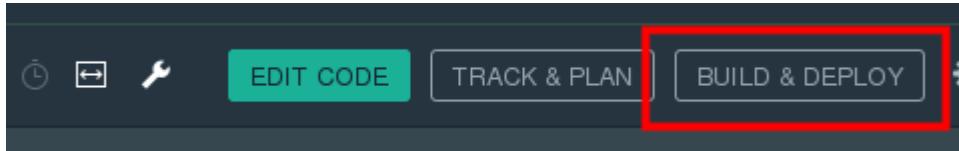
4. Introduce a change to the project's code by editing the public/index.html file and alter the <h1> header tag to something unique.

index.html

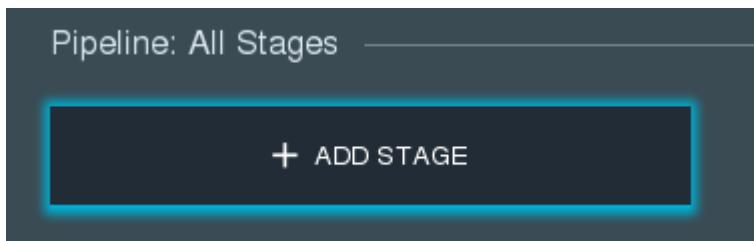
```
1  <!DOCTYPE html>
2  <html>
3
4      <head>
5          <title>NodeJS Starter Application</title>
6          <meta charset="utf-8">
7          <meta http-equiv="X-UA-Compatible" content="IE=edge">
8          <meta name="viewport" content="width=device-width, in
9              <link rel="stylesheet" href="stylesheets/style.css">
10             </head>
11
12         <body>
13             <table>
14                 <tr>
15                     <td style= "width:30%;">
16                         NodeJS Starter Application
21                                     Get started by reading our
22                                         <a href = "https://www.ng.bluemix.net/docs/#s
23                                         or use the Start Coding guide under your app
24
25             </table>
26         </body>
27     </html>
```

## Exercise 5.5.2: Configuring the Build and Deploy pipeline

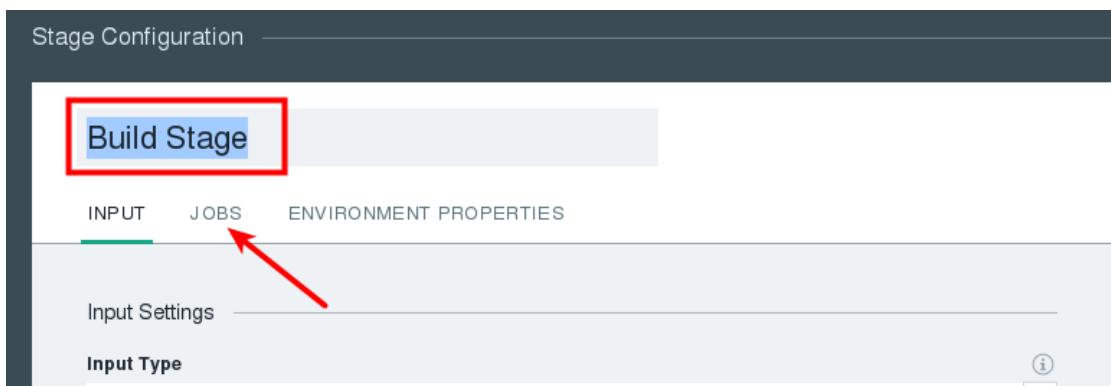
1. Switch to the pipeline overview by clicking **BUILD & DEPLOY**.



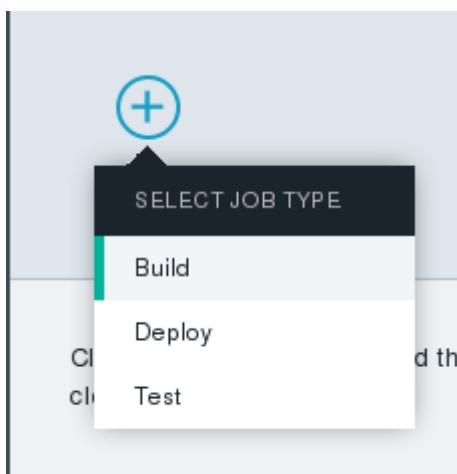
2. Click **ADD STAGE**.



3. Name the stage **Build Stage** and then click the **JOB TAB**.



4. Click **ADD A JOB > Build**.



## Exercise 5.5: Configuring a continuous delivery pipeline

5. Change Builder Type to `npm`. Then, scroll to the bottom and click **SAVE**.

6. In the **Pipeline: All Stages** view, click **ADD STAGE** again.

The screenshot shows the 'Pipeline: All Stages' interface. On the left, there is a card for a 'Build Stage'. The stage is labeled 'STAGE NOT RUN'. It has a 'LAST INPUT' section showing 'Not yet run' and a 'JOBS' section with a single entry 'Build' which is also 'Not yet run'. Below the stage card is a 'LAST EXECUTION RESULT' section stating 'No results'. On the right side of the screen, there is a large, empty rectangular area with a red border around its top edge. Inside this area, at the top center, is a button labeled '+ ADD STAGE'.

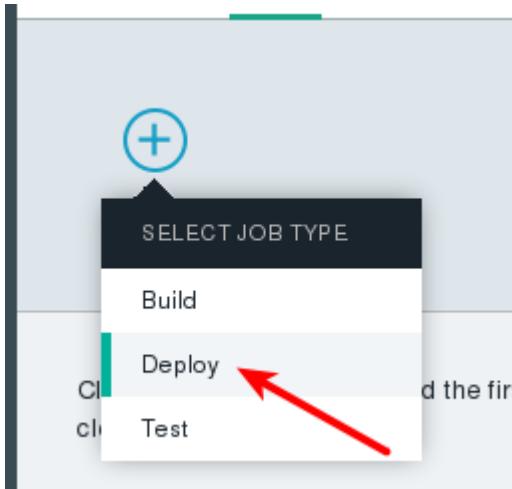
7. Name the stage **Deploy Stage** and note the options available for the stage. The default configuration will use the artifacts that were created from the Build Job in the Build Stage as input, and the Deploy Stage jobs will run after successful completion of the previous stage (Build Stage).

The screenshot shows the 'Stage Configuration' interface for a 'Deploy Stage'. At the top, the stage name 'Deploy Stage' is displayed in a blue header bar, which is highlighted with a red box. Below the stage name, there are three tabs: 'INPUT' (underlined in green), 'JOBS', and 'ENVIRONMENT PROPERTIES'. A red arrow points from the stage name down towards the 'JOBS' tab. Under the 'INPUT' tab, there is a section titled 'Input Settings' and a subsection titled 'Input Type'.

8. Click the **JOBS** tab.

## Exercise 5.5: Configuring a continuous delivery pipeline

9. Click **ADD JOB > Deploy**.



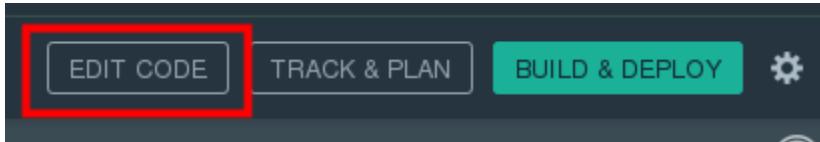
The Deploy stage will default to acceptable options, but take a moment to view the different settings that are available. The default settings are gathered from the configuration that you entered when you first forked the project.

10. Accept the default settings and click **SAVE**. Observe that both stages are now configured in the pipeline.

A screenshot of a pipeline configuration interface titled "Pipeline: All Stages". The interface shows two stages: "Build Stage" and "Deploy Stage". Both stages are currently "STAGE NOT RUN".  
  
Build Stage:  
- LAST INPUT: Not yet run  
- JOBS: Build (Not yet run)  
- LAST EXECUTION RESULT: No results  
  
Deploy Stage:  
- LAST INPUT: Stage: Build Stage / Job: B...  
- JOBS: Deploy (Not yet run)  
- LAST EXECUTION RESULT: No results

### Exercise 5.5.3: Triggering the pipeline by pushing to the Git repository

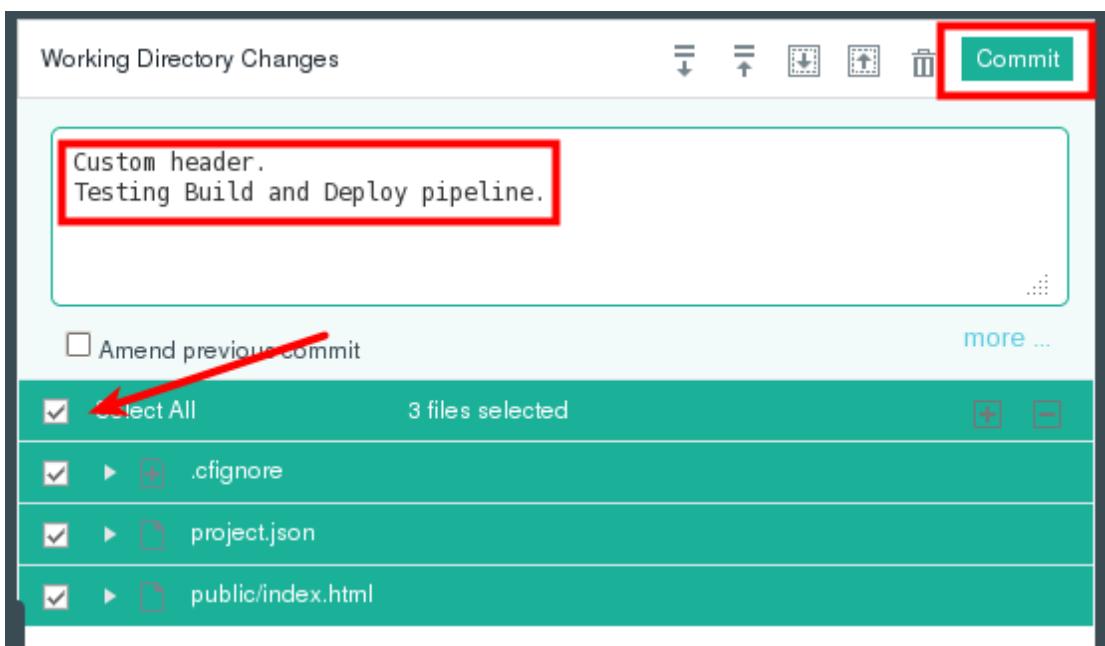
1. Click **EDIT CODE**.



2. Click the **Git** icon in the left column.



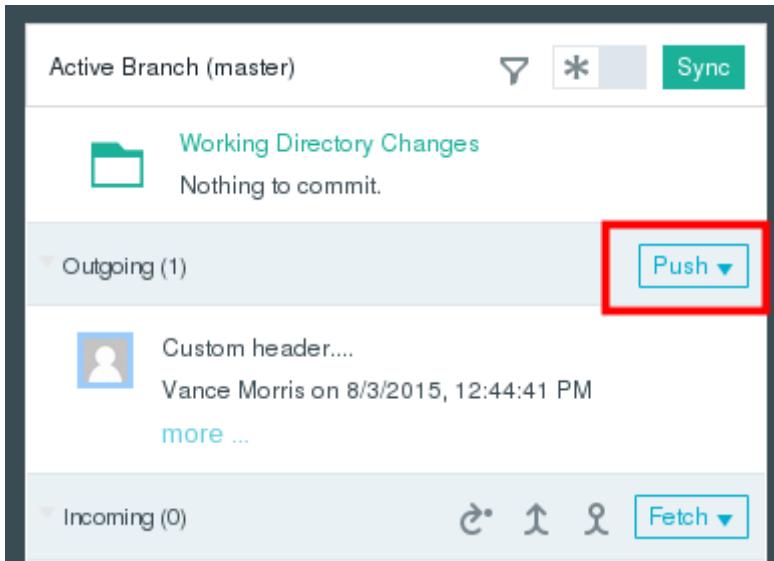
3. Enter an informative commit message and select the **Select All** check box. Then, click **Commit**.



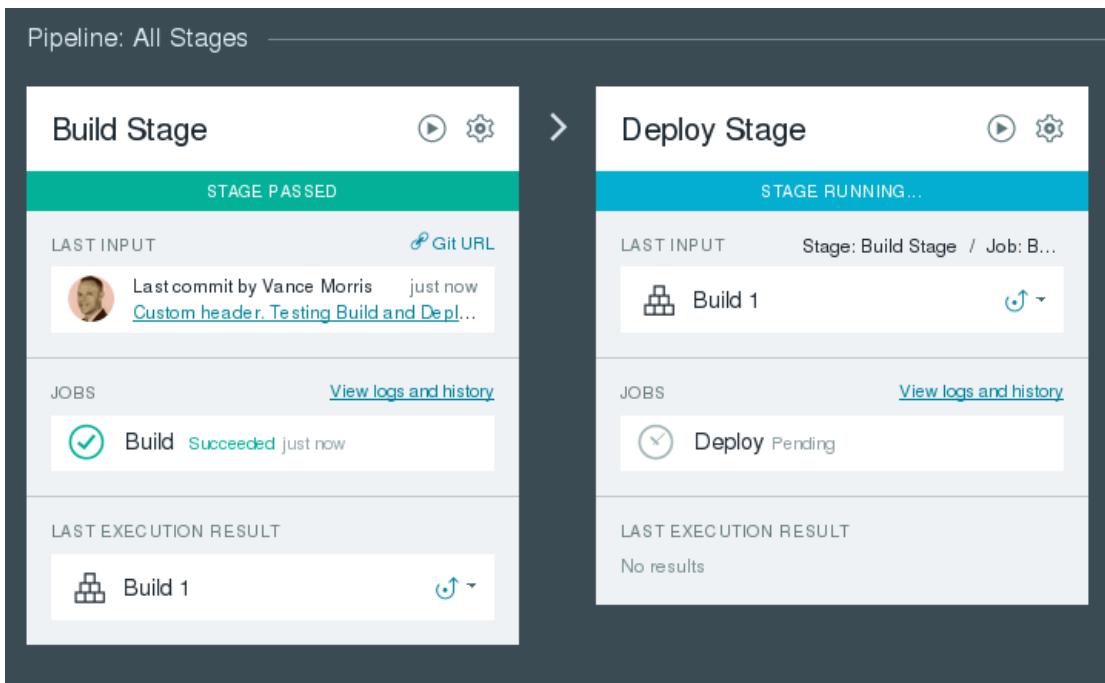
## Exercise 5.5: Configuring a continuous delivery pipeline

Note that the outgoing commit is now staged.

4. Click **PUSH** and then immediately click **BUILD & DEPLOY**.

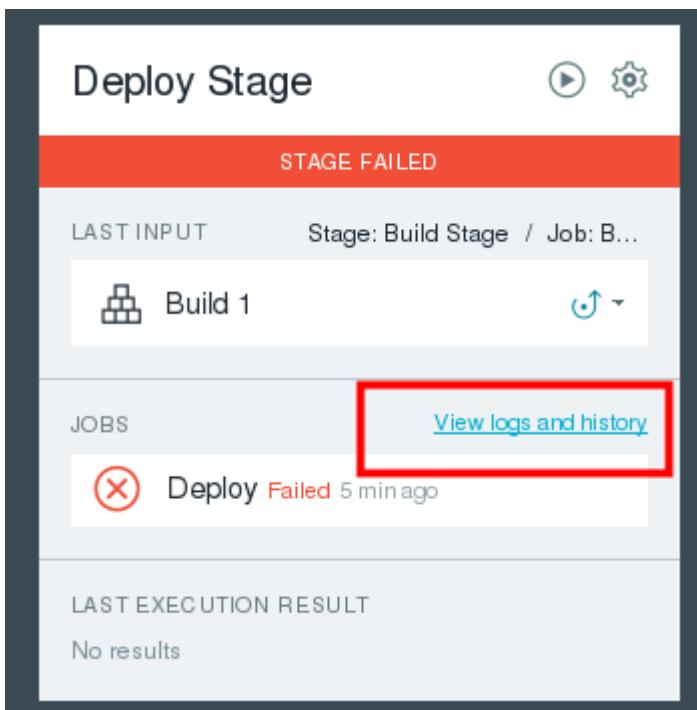


Observe how the build stage is automatically started and once complete, the deploy stage is automatically started.



## Exercise 5.5: Configuring a continuous delivery pipeline

5. The deploy stage should fail. To determine why, click **View logs and history**.



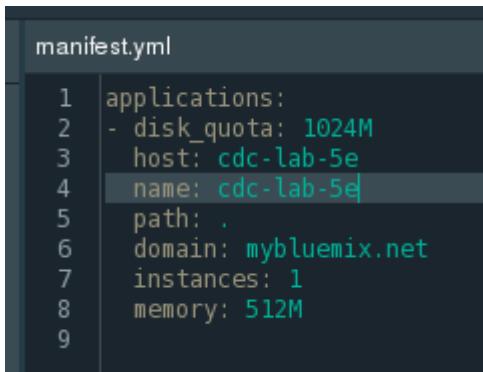
6. From the log, note that Cloud Foundry determined that requested route "REPLACE WITH CUSTOM NAME.mybluemix.net" is not valid.

```
target: https://api.ng.bluemix.net
+ source _deploy.sh
++ cf push cdc-lab-5e
Using manifest file /home/jenkins/workspace/c14e6643-a29f-0a3b-d44e-
Creating app cdc-lab-5e in org vmorris@us.ibm.com / space dev as vmor
OK

Creating route REPLACE WITH CUSTOM NAME.mybluemix.net...
FAILED
Server error, status code: 400, error code: 210001, message: The ro
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

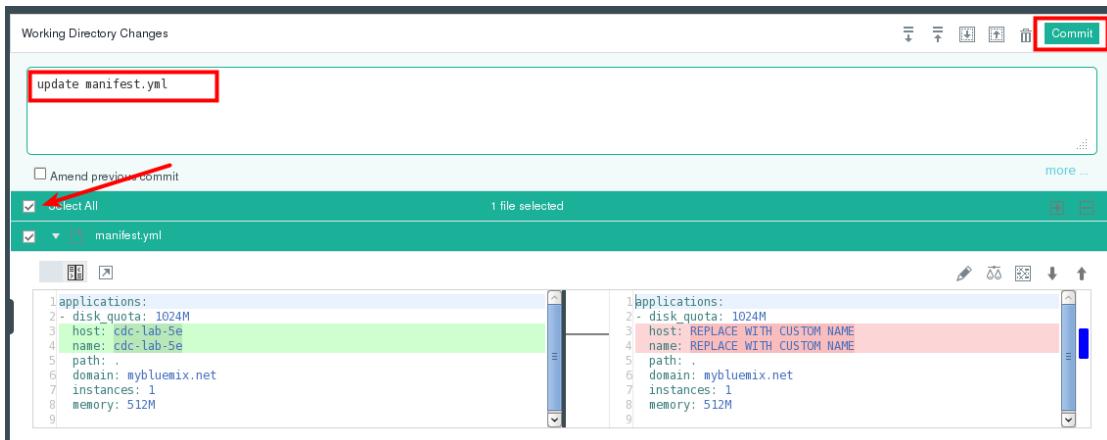
### Exercise 5.5: Configuring a continuous delivery pipeline

7. Return to the web GUI code editor and open the manifest.yml file. Replace the host and name values with a unique name that will be used when you create the route and URL to your application.



```
manifest.yml
1 applications:
2 - disk_quota: 1024M
3 host: cdc-lab-5e
4 name: cdc-lab-5e
5 path: .
6 domain: mybluemix.net
7 instances: 1
8 memory: 512M
9
```

8. Open the Git repository view and commit the changed manifest file.



9. Just as you did before, push the staged outgoing commit and then switch back to the BUILD & DEPLOY view. Observe the Build and Deploy stages execution again.

## Exercise 5.5: Configuring a continuous delivery pipeline

The screenshot shows a pipeline interface with two stages: Build Stage and Deploy Stage.

**Build Stage:**

- STAGE PASSED**
- LAST INPUT:** Last commit by Vance Morris just now ([updated manifest.yml](#))
- JOB:** Build **Succeeded** just now
- LAST EXECUTION RESULT:** Build 2

**Deploy Stage:**

- STAGE RUNNING...**
- LAST INPUT:** Stage: Build Stage / Job: B...
- JOB:** Deploy **Running** 39%
- LAST EXECUTION RESULT:** No results

#### Exercise 5.5: Configuring a continuous delivery pipeline

10. During the deploy stage's execution, click **View logs and history** and scroll to the bottom of the log stream. After the application has successfully deployed to Bluemix, you will see a Finished: SUCCESS message.

```
urls: cdc-lab-5e.mybluemix.net
last uploaded: Mon Aug 3 18:03:44 UTC 2015

      state      since          cpu    memory      disk
#0  running  2015-08-03 06:05:06 PM  0.0%   73.4M of 512M  54.4M of 1G
  Sending deployment success of cdc-lab-5e to IBM DevOps Services...
  IBM DevOps Services notified successfully.
  Oct 1vw
Finished: SUCCESS
```

11. Scroll to the top of the page and click **Back to Pipeline**.

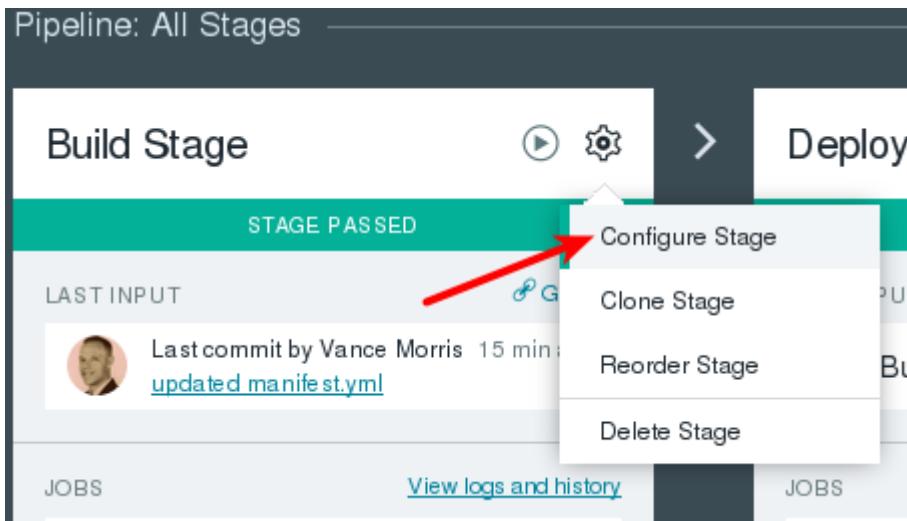


12. Locate the Last Execution Result section inside the Deploy stage summary and note that the status of the Bluemix runtime is displayed along with a hyperlink to open the application. Click that link.

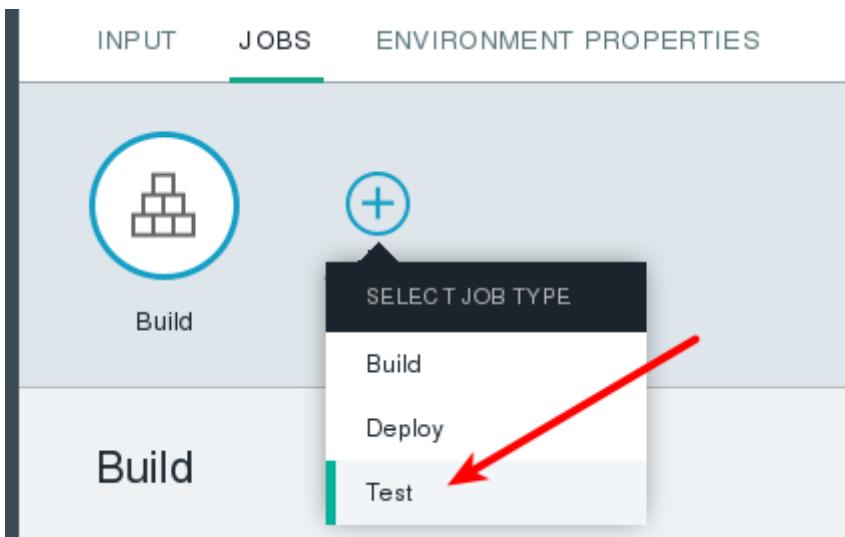
A screenshot of a web-based deployment interface. At the top, it says "Deploy Stage" with a play button and gear icon. Below that is a green bar labeled "STAGE PASSED". Underneath, there's a "LAST INPUT" section with a "Build 2" icon and a dropdown arrow. Then there's a "JOBS" section with a green checkmark icon, the text "Deploy Succeeded 2 min ago", and a "View logs and history" link. Finally, there's a "LAST EXECUTION RESULT" section. It shows a "cdc-lab-5e" icon with a rocket ship, a link "cdc-lab-5e.mybluemix.net" which is underlined and highlighted with a red arrow pointing to it, and a "View runtime log" link. Below this is another "Build 2" icon and a dropdown arrow.

## Exercise 5.5.4: Configuring a test job and modifying the pipeline

1. Return to the DevOps Services Build and Deploy Pipeline overview and from the Build Stage section, click **Stage Configuration**. Then, click **Configure Stage**.



2. Click **ADD JOB > Test**.



3. Accept the default configuration for the Test job. Note that the Simple Tester Type allows you to execute a custom Bash script. Our simple project does not have any testing framework to execute, but if it did, this is the place to execute and evaluate tests. Click **SAVE**. Note that a new job is added to the Build Stage overview.

## Exercise 5.5: Configuring a continuous delivery pipeline

The screenshot shows the 'Build Stage' interface. At the top, it says 'STAGE PASSED'. Below that, under 'LAST INPUT', there is a profile picture of a man and the text 'Last commit by Vance Morris 18 min ago [updated manifest.yml](#)'. Under 'JOBS', there are two entries: 'Build' (status: Succeeded, 17 min ago) and 'Test' (status: Not run). A red box highlights the 'Test' entry. Below that, under 'LAST EXECUTION RESULT', it shows 'Build 2' with a download icon.

4. Manually start the Build stage by clicking the play button at the top.

The screenshot shows the 'Build Stage' interface. The play button at the top is highlighted with a red box.

5. Note that the Build job executes, followed by the Test job. If either of these jobs were to fail, the pipeline execution would stop and the Deploy Stage would not execute.

The screenshot shows the 'Build Stage' interface. Under 'JOBS', there are two entries: 'Build' (status: Succeeded, just now) and 'Test' (status: Succeeded, just now). Both entries have green checkmarks.