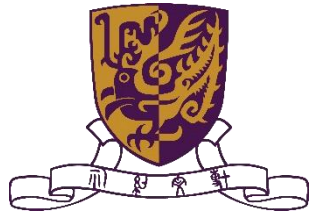


Tutorial 2 – OpenShift & CGI Programming

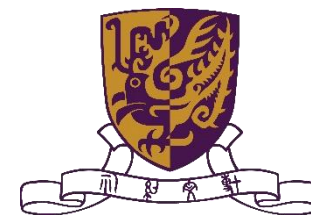
CSCI 4140: Open-Source Software Project Development Spring 2018

WANG, Yue
01/25/2018



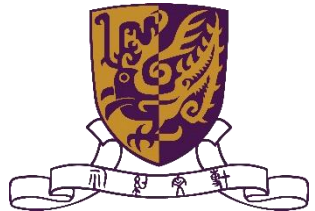
Outlines

- OpenShift v3 deployment
- Basic Python CGI programming



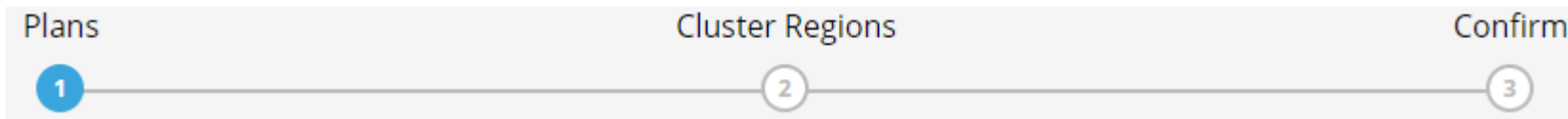
Background

- RedHatCloud Service
 - <https://www.openshift.com/>
 - Platform as a Service (PaaS)
 - **Openshift v3** (many differe)
- Free (starter plan)
 - Up to one project
 - 1GiB of persistent storage
 - 1GiB of memory
- Setup web 'server' easily
 - Support perl, PHP, node.js, ruby etc.
 - One click to setup database

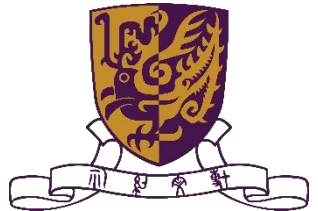
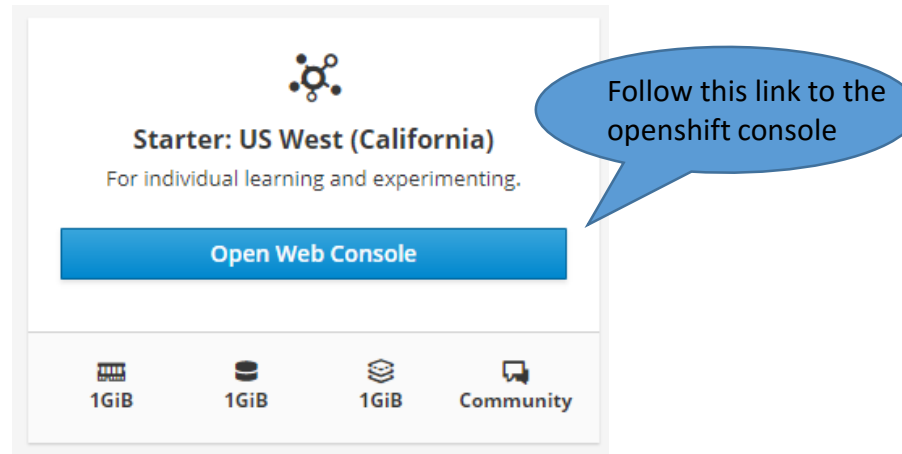


Get started

- Create an account for openshift.com
- Then go to here (<https://manage.openshift.com/register/plan>) to choose a plan (starter plan)



- After done, you will get this:

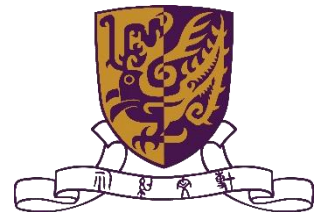


Web Console

The screenshot shows the OpenShift Online web console. At the top, the header reads "OPENSIFT ONLINE" (note the typo in the image). On the right of the header, there is a notification for "1 open issue", a help icon, and a user profile for "yue wang". Below the header is a search bar labeled "Search Catalog". The main content area is titled "Browse Catalog" and includes tabs for "All", "Languages", "Databases", "Middleware", "CI/CD", and "Other". A filter dropdown shows "28 Items". The catalog displays various project templates in a grid, including .NET Core Builder Images, Apache HTTP Server (httpd), CakePHP + MySQL (Persistent), Docker + MySQL (Persistent), Django + PostgreSQL (Persistent), Jenkins (Persistent), Laravel + MySQL (Persistent), MariaDB (Persistent), MongoDB (Persistent), MySQL (Persistent), Node.js, Node.js + MongoDB (Persistent), Perl, PHP, and PostgreSQL (Persistent). On the right sidebar, the "My Projects" section shows "1 of 1 Projects" with a "View All" link and a "Create Project" button. Below this is a "Getting Started" section with a "Take Home Page Tour" button and links to Documentation, Interactive Learning Portal, Container Development Kit, YouTube, and Blog. A "Recently Viewed" section shows icons for Python and Node.js + MongoDB (Persistent).

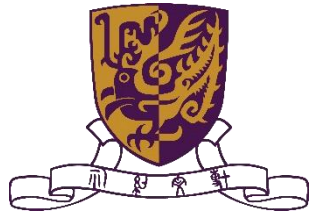
Create your own project here

https://docs.openshift.com/online/getting_started/basic_walkthrough.html



Web Console

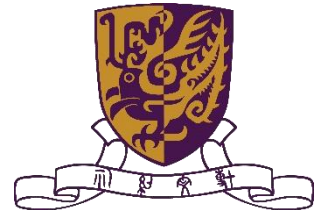
- The OpenShift Online web console is a user interface accessible from a web browser. Developers can use the web console to visualize, browse, and manage the contents of [projects](#).
- Another way to manage your project
 - **Command Line Tools (CLI)**
 - Download it from here (<https://console.starter-us-west-1.openshift.com/console/command-line>)
 - E.g. `$ oc new-project <project_name>`



Project Overviews

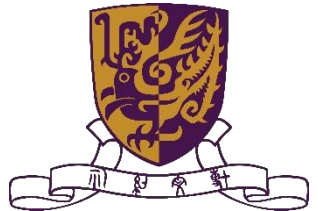
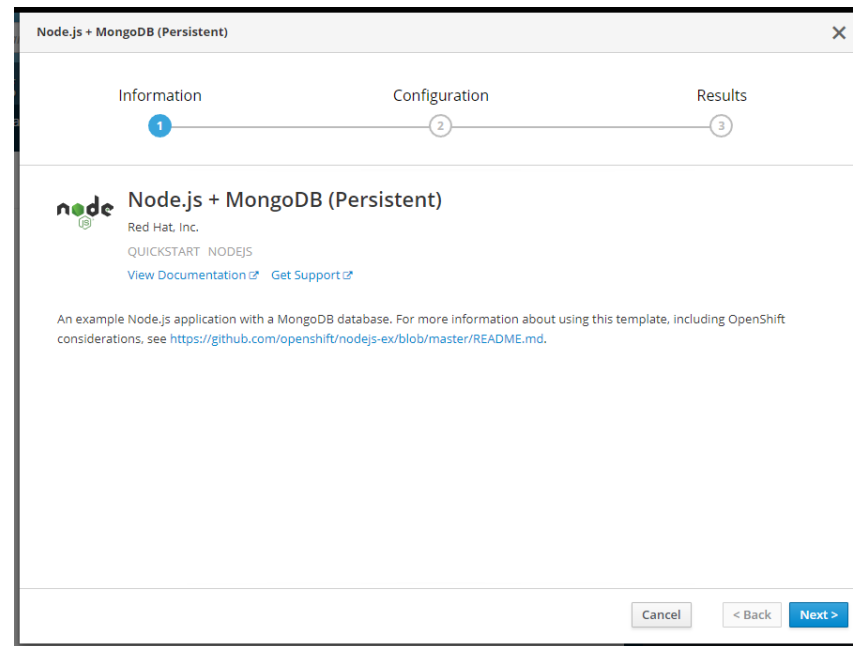
The screenshot shows the OpenShift Project Overview page for a project named 'Hello Openshift'. The left sidebar contains navigation tabs: Overview (selected), Applications, Builds, Resources, Storage, and Monitoring. The main content area displays two project sections: 'NODEJS' and 'RAILS POSTGRESQL EXAMPLE'. Each section shows a build status (e.g., 'Build nodejs, #1' is 'Complete'), deployment details (e.g., 'Deployment: nodejs - 7 minutes ago'), and a high-level view of components (e.g., '1 pod' for nodejs, '3 pods' for postgresql). The interface includes a top navigation bar with a home icon, project selector, and user profile.

1. The project selector allows you to switch between projects you have access to.
2. Create new applications using a source repository or using a template.
3. The Overview tab (currently selected) visualizes the contents of your project with a high-level view of each component.
4. Applications tab: Browse and perform actions on your deployments, pods, services, and routes.
5. Builds tab: Browse and perform actions on your builds and image streams.
6. Resources tab: View your current quota consumption and other resources.
7. Storage tab: View persistent volume claims and request storage for your applications.
8. Monitoring tab: View logs for builds, pods, and deployments, as well as event notifications for all objects in your project.



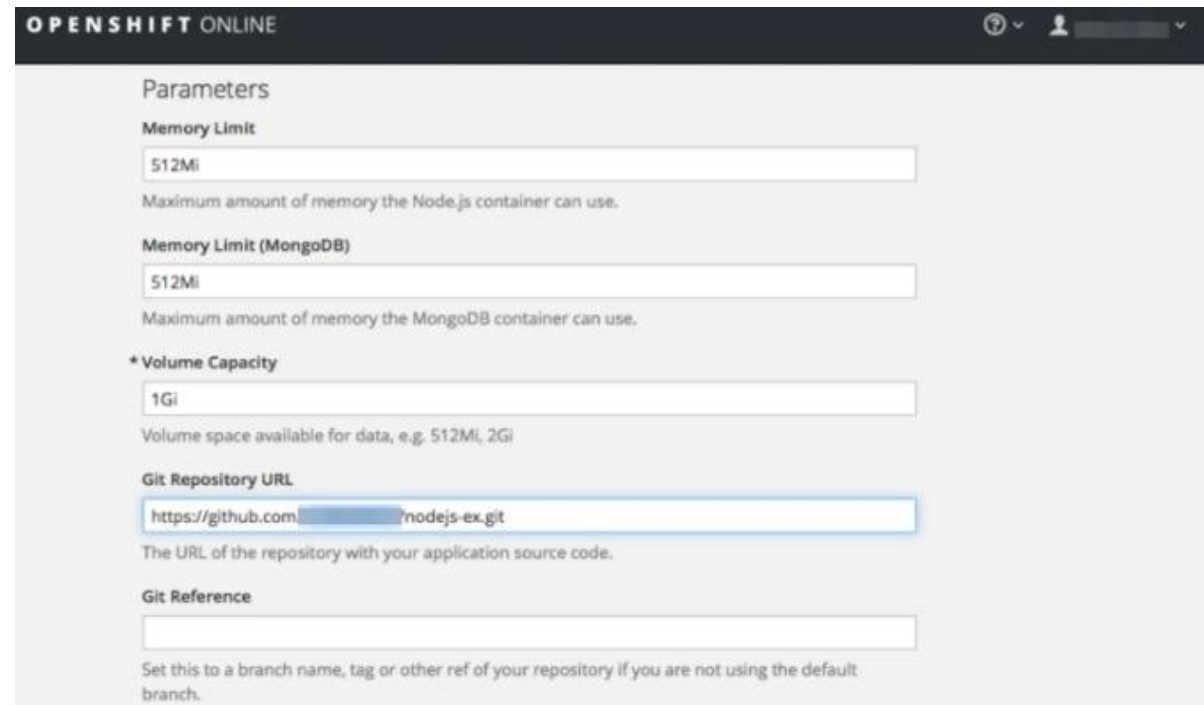
Create your first project

- Press the create project button
- Edit the project specification
- Deploy applications on your project, e.g.



Connect your github with openshift project

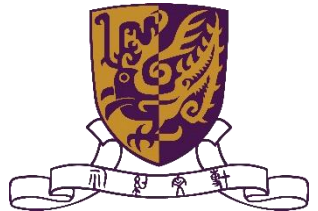
- You need to place all your codes in your github
- When configuring the selected application, add your git repo



The screenshot shows the 'Parameters' section of the OpenShift Online interface. It includes the following fields and descriptions:

- Memory Limit:** A text input field containing '512Mi'. Below it, the text reads: 'Maximum amount of memory the Node.js container can use.'
- Memory Limit (MongoDB):** A text input field containing '512Mi'. Below it, the text reads: 'Maximum amount of memory the MongoDB container can use.'
- * Volume Capacity:** A text input field containing '1Gi'. Below it, the text reads: 'Volume space available for data, e.g. 512Mi, 2Gi'
- Git Repository URL:** A text input field containing 'https://github.com/[redacted]/nodejs-ex.git'. Below it, the text reads: 'The URL of the repository with your application source code.'
- Git Reference:** An empty text input field. Below it, the text reads: 'Set this to a branch name, tag or other ref of your repository if you are not using the default branch.'

- Finally, click Create to deploy your application.



Configuring Automated Builds

- GitHub webhook: automatically trigger a rebuild of your application whenever you push code changes to your forked repository.
- In the OpenShift Online web console:
 - Navigate to the project containing your application.
 - Click the Browse tab, then click Builds, then click the name of the build for your Node.js application.
 - From the Configuration tab, click copy next to GitHub webhook URL to copy your GitHub webhook.

nodejs-mongo-persistent created 4 days ago Start Build Actions ▾

app nodejs-mongo-persistent template nodejs-mongo-persistent

History Configuration Environment Events

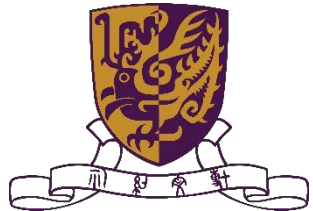
Details

Build Strategy:	Source
Source Repo:	https://github.com/ayueei/nodejs-ex.git
Builder Image:	openshift/nodejs:6
Output To:	csci4140/nodejs-mongo-persistent:latest
Run Policy:	Serial ⓘ

Post-Commit Hooks

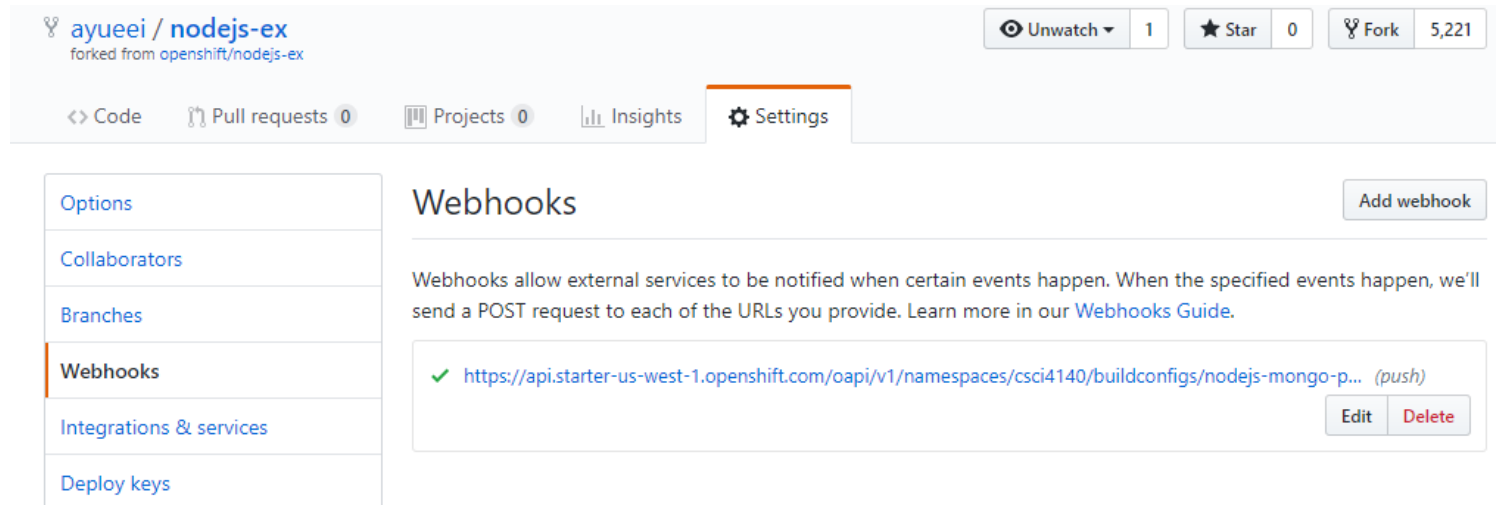
Triggers [Learn More](#) ⓘ

Config Change For:	Build config nodejs-mongo-persistent
New Image For:	openshift/nodejs:6
Generic Webhook URL:	https://api.starter-us-west-1.openshift.com ⓘ
GitHub Webhook URL:	https://api.starter-us-west-1.openshift.com ⓘ
Manual (CLI):	<code>oc start-build nodejs-mongo-persistent -n c</code> ⓘ

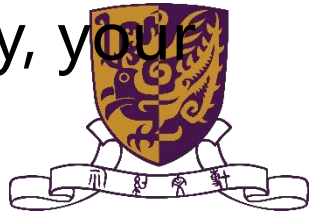


Configuring Automated Builds

- Paste the copied payload URL in github
- Choose the content type as **application/json**
- After done, you will see:



- The next time you push a code change to your forked repository, your application will automatically rebuild.



Viewing Your Running Application

- Follow the link in the Overview page

OPENSIFT ONLINE

1 open issue

yue wang

csci4140

Add to Project

Overview

Applications

Builds

Resources

Storage

Monitoring

Name Filter by name

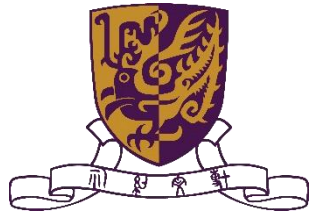
List by Application

APPLICATION

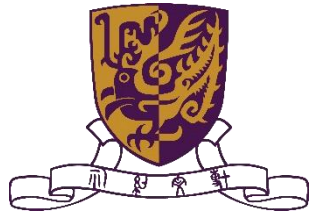
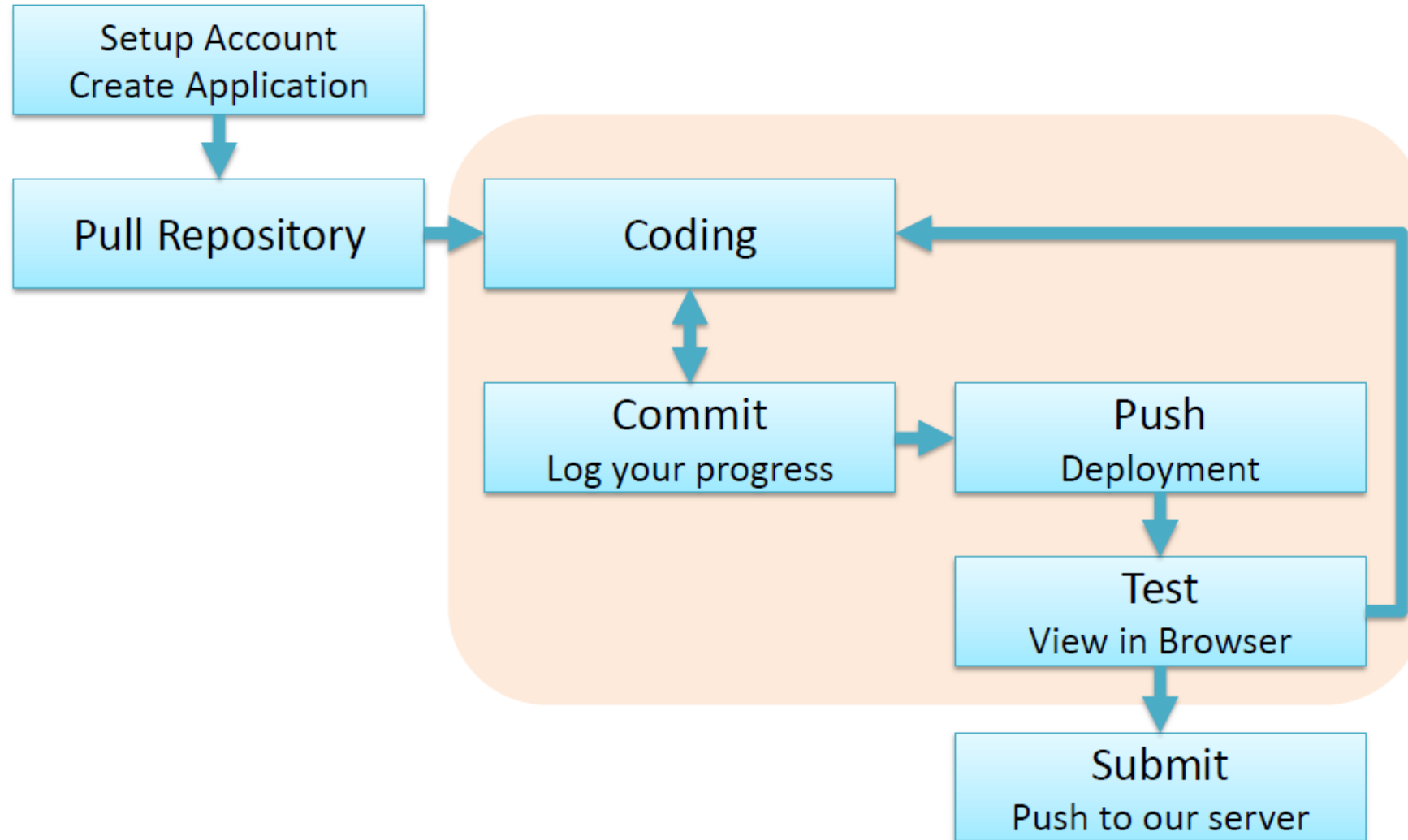
nodejs-mongo-persistent

<http://nodejs-mongo-persistent-csci4140.a3c1.starter-us-west-1.openshiftapps.com>

	DEPLOYMENT	Mib Memory	Cores CPU	Kib/s Network	
>	mongodb, #1	--	--	--	1 pod
>	nodejs-mongo-persistent, #2	--	--	--	1 pod



Develop using OpenShift



A case study

- Fork <https://github.com/openshift/nodejs-ex> into your github repo and then follow the previous processes to create a project using two applications (Mongodb and Nodejs) .
- After clicking the generated link, you will see the page like:

Welcome to your Node.js application on OpenShift

How to use this example application

For instructions on how to use this application with OpenShift, start by reading the [Developer Guide](#).

Managing your application

Documentation on how to manage your application from the Web Console or Command Line is available at the [Developer Guide](#).

Deploying code changes

The source code for this application is available to be forked from the [OpenShift GitHub repository](#). You can configure a webhook in your repository to make OpenShift automatically start a build whenever you push your code:

1. From the Web Console homepage, navigate to your project
2. Click on Browse > Builds
3. From the view for your Build click on the button to copy your GitHub webhook
4. Navigate to your repository on GitHub and click on repository settings > webhooks
5. Paste your webhook URL provided by OpenShift — that's it!

After you save your webhook, if you refresh your settings page you can see the status of the ping that Github sent to OpenShift to verify it can reach the server.

Web Console

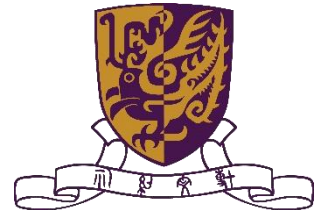
You can use the Web Console to view the state of your application components and launch new builds.

Command Line

With the [OpenShift command line interface \(CLI\)](#), you can create applications and manage projects from a terminal.

Development Resources

- [OpenShift Documentation](#)
- [OpenShift Origin GitHub](#)
- [Source To Image GitHub](#)
- [Getting Started with Node.js on OpenShift](#)
- [Stack Overflow questions for OpenShift](#)
- [Git documentation](#)



A case study

- Then you set the webhook in the github to openshift.
- You revise the code (index.html) in the local repo to modify the title (from “welcome to your Node.js” into “This is awesome Node.js”)
- Commit the change and push to the remote server.
- Refresh your page and you will get:

This is awesome Node.js application on OpenShift

How to use this example application

For instructions on how to use this application with OpenShift, start by reading the [Developer Guide](#).

Deploying code changes

The source code for this application is available to be forked from the [OpenShift GitHub repository](#). You can configure a webhook in your repository to make OpenShift automatically start a build whenever you push your code:

1. From the Web Console homepage, navigate to your project
2. Click on Browse > Builds
3. Click the link with your BuildConfig name
4. Click the Configuration tab
5. Click the "Copy to clipboard" icon to the right of the "GitHub webhook URL" field
6. Navigate to your repository on GitHub and click on repository settings > webhooks > Add webhook
7. Paste your webhook URL provided by OpenShift
8. Leave the defaults for the remaining fields — that's it!

After you save your webhook, if you refresh your settings page you can see the status of the ping that Github sent to OpenShift to verify it can reach the server.

Managing your application

Documentation on how to manage your application from the Web Console or Command Line is available at the [Developer Guide](#).

Web Console

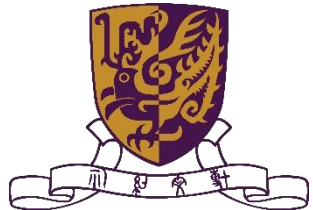
You can use the Web Console to view the state of your application components and launch new builds.

Command Line

With the [OpenShift command line interface](#) (CLI), you can create applications and manage projects from a terminal.

Development Resources

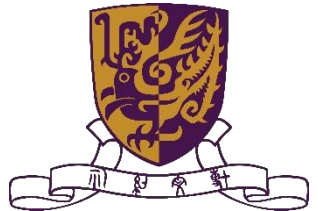
- [OpenShift Documentation](#)
- [OpenShift Origin GitHub](#)
- [Source To Image GitHub](#)
- [Getting Started with Node.js on OpenShift](#)
- [Stack Overflow questions for OpenShift](#)
- [Git documentation](#)



Core concept of openshift v3

- **Containers and images** are the building blocks for deploying your applications.
- **Pods and services** allow for containers to communicate with each other and proxy connections.
- **Projects and users** provide the space and means for communities to organize and manage their content together.
- **Builds and image streams** allow you to build working images and react to new images.
- **Deployments** add expanded support for the software development and deployment lifecycle.
- **Routes** announce your service to the world.
- **Templates** allow for many objects to be created at once based on customized parameters.

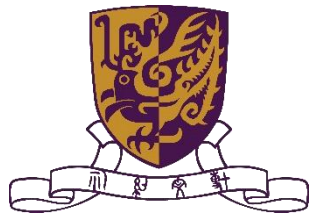
https://docs.openshift.com/online/architecture/core_concepts/index.html



Python - CGI Programming

- What is CGI?
 - The Common Gateway Interface, or CGI, is a standard for external gateway programs to interface with information servers such as HTTP servers.
- Web Browsing
 - Your browser contacts the HTTP web server and demands for the URL, i.e., filename.
 - Web Server parses the URL and looks for the filename. If it finds that file then sends it back to the browser, otherwise sends an error message indicating that you requested a wrong file.
 - Web browser takes response from web server and displays either the received file or error message.

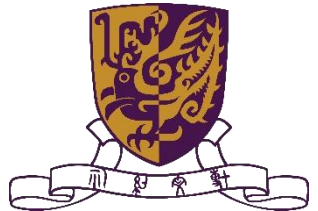
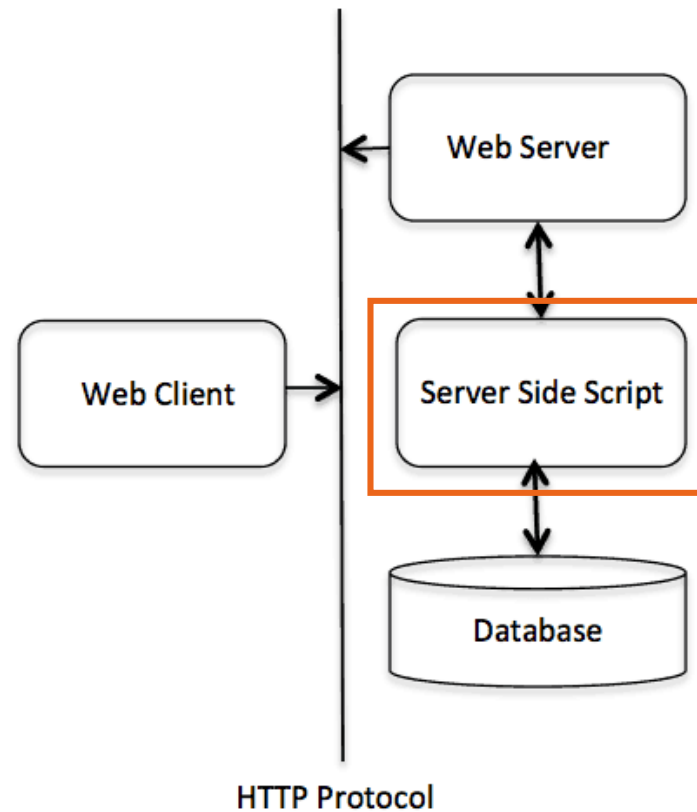
https://www.tutorialspoint.com/python/python_cgi_programming.htm



Python - CGI Programming

- These CGI programs can be a Python Script, PERL Script, Shell Script, C or C++ program, etc

CGI Architecture Diagram

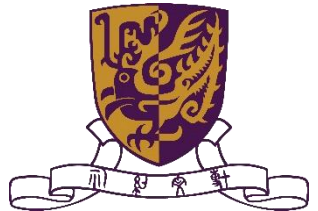


Python - CGI Programming

- Web Server Support and Configuration
 - Store your scripts under /var/www/cgi-bin in your server (default)
- First CGI Program

```
#!/usr/bin/python  
  
print "Content-type:text/html\r\n\r\n"  
print '<html>'  
print '<head>'  
print '<title>Hello Word - First CGI Program</title>'  
print '</head>'  
print '<body>'  
print '<h2>Hello Word! This is my first CGI program</h2>'  
print '</body>'  
print '</html>'
```

HTTP header which is sent to the browser to understand the content



Python - CGI Programming

- Passing Information using GET method
 - The GET method sends the encoded user information appended to the page request.
 - E.g. `http://www.test.com/cgi-bin/hello.py?key1=value1&key2=value2`
 - An simple example:

```
#!/usr/bin/python

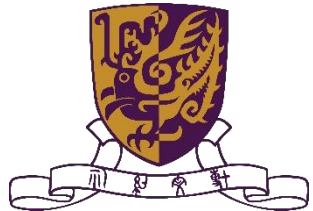
# Import modules for CGI handling
import cgi, cgitb

# Create instance of FieldStorage
form = cgi.FieldStorage()

# Get data from fields
first_name = form.getvalue('first_name')
last_name = form.getvalue('last_name')

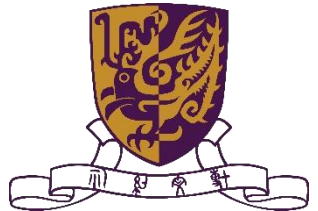
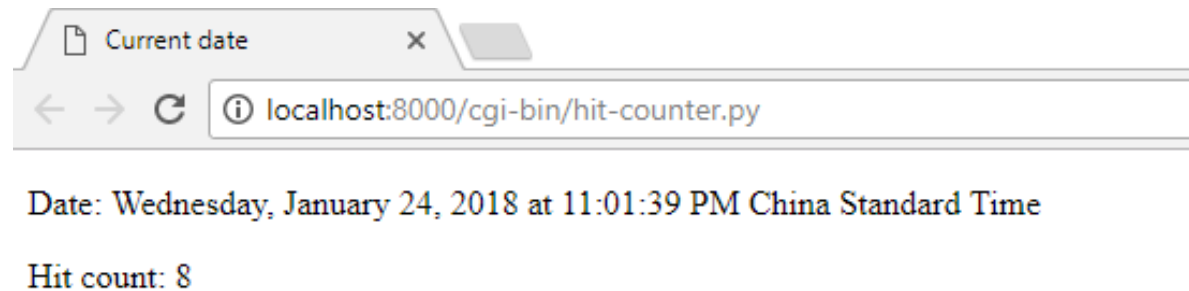
print "Content-type:text/html\r\n\r\n"
print "<html>"
print "<head>"
print "<title>Hello - Second CGI Program</title>"
print "</head>"
print "<body>"
print "<h2>Hello %s %s</h2>" % (first_name, last_name)
print "</body>"
print "</html>"
```

Output: Hello ZARA ALI



An example-hit counter

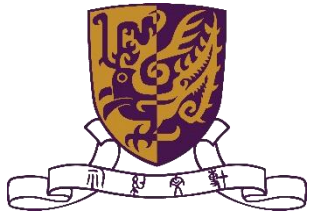
- Fork the repo (<https://github.com/ayueei/python-cgi-example>) into your github
- Run the script http-server.py in your local machine (use python2), you will get:



An example-hit counter

- Calculate the hit count and write to a file

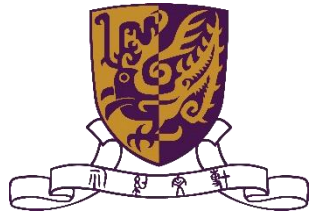
```
3 import cgi
4 import cgitb
5 import time
6 import os
7 cgitb.enable()
8
9 hit_count_path = os.path.join(os.path.dirname(__file__), "hit-count.txt")
10
11 ▼ if os.path.isfile(hit_count_path):
12     hit_count = int(open(hit_count_path).read())
13     hit_count += 1
14 else:
15     hit_count = 1
16
17 hit_counter_file = open(hit_count_path, 'w')
18 hit_counter_file.write(str(hit_count))
19 hit_counter_file.close()
```



An example-hit counter

- Return a html header with a html file

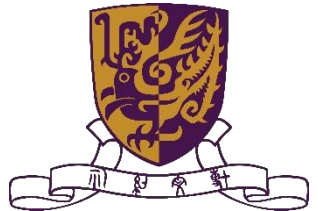
```
21 header = "Content-type: text/html\n\n"
22 date_string = time.strftime('%A, %B %d, %Y at %I:%M:%S %p %Z')
23
24 html = ""
25 <!DOCTYPE html>
26 <html lang="en">
27 <head>
28     <meta charset="utf-8">
29     <title>Current date</title>
30 </head>
31 <body>
32     <p>
33         Date: {0}
34     </p>
35     <p>
36         Hit count: {1}
37     </p>
38 </body>
39 </html>
40 """.format(cgi.escape(date_string), cgi.escape(str(hit_count)))
41
42 print header + html
```



An example-hit counter

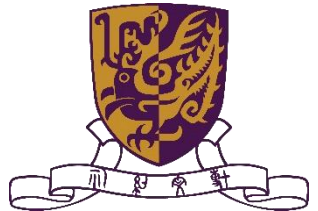
- Set up a local http server and trigger the scripts:

```
7  import BaseHTTPServer
8  import CGIHTTPServer
9  import webbrowser
10
11  PORT = 8000
12  #TODO: check that port is available,
13  # and look for a different one if it isn't.
14
15  script_path = "cgi-bin/hit-counter.py"
16
17  server_class = BaseHTTPServer.HTTPServer
18  handler_class = CGIHTTPServer.CGIHTTPRequestHandler
19  server_address = ("", PORT)
20
21  httpd = server_class(server_address, handler_class)
22
23  url = 'http://localhost:{0}/{1}'.format(PORT, script_path)
24
25  webbrowser.open_new_tab(url)
26
27  print("serving at", url)
28
29  httpd.serve_forever()
```



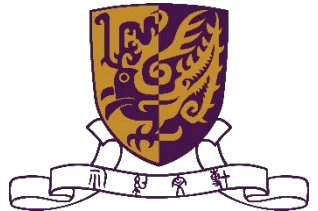
Things to do

- How to set up this hit counter in your openshift server?
- How to interact with database in a python cgi program?
- More advanced python cgi programming...



Some useful links

- https://www.tutorialspoint.com/openshift/openshift_getting_started.htm
- <https://docs.openshift.com/online/welcome/index.html>
- https://www.tutorialspoint.com/python/python_cgi_programming.htm



Thanks for listening !

