



COMP4442 Service and Cloud Computing

Lab 6



Outline

- Review of Lab5
- How to develop web app to visualize real-time data
 - Store data constantly to the database
 - Build a HTML chart with JavaScript
 - Create web app to visual real-time data with flask
 - Deploy code in AWS Beanstalk



Review of Lab5

- We learned how to use AWS Spark to count the frequency of words in large file.
- But how to visual the analytical results, e.g., in a webpage

```
import os
import sys

from pyspark import SparkContext

args = sys.argv
inp = args[1]
out = args[2]

sc = SparkContext()

text_file = sc.textFile(inp)
counts = text_file.flatMap(lambda line: line.split(" ")) \
    .map(lambda word: (word, 1)) \
    .reduceByKey(lambda a, b: a + b)
counts.saveAsTextFile(out)

sc.stop()
```

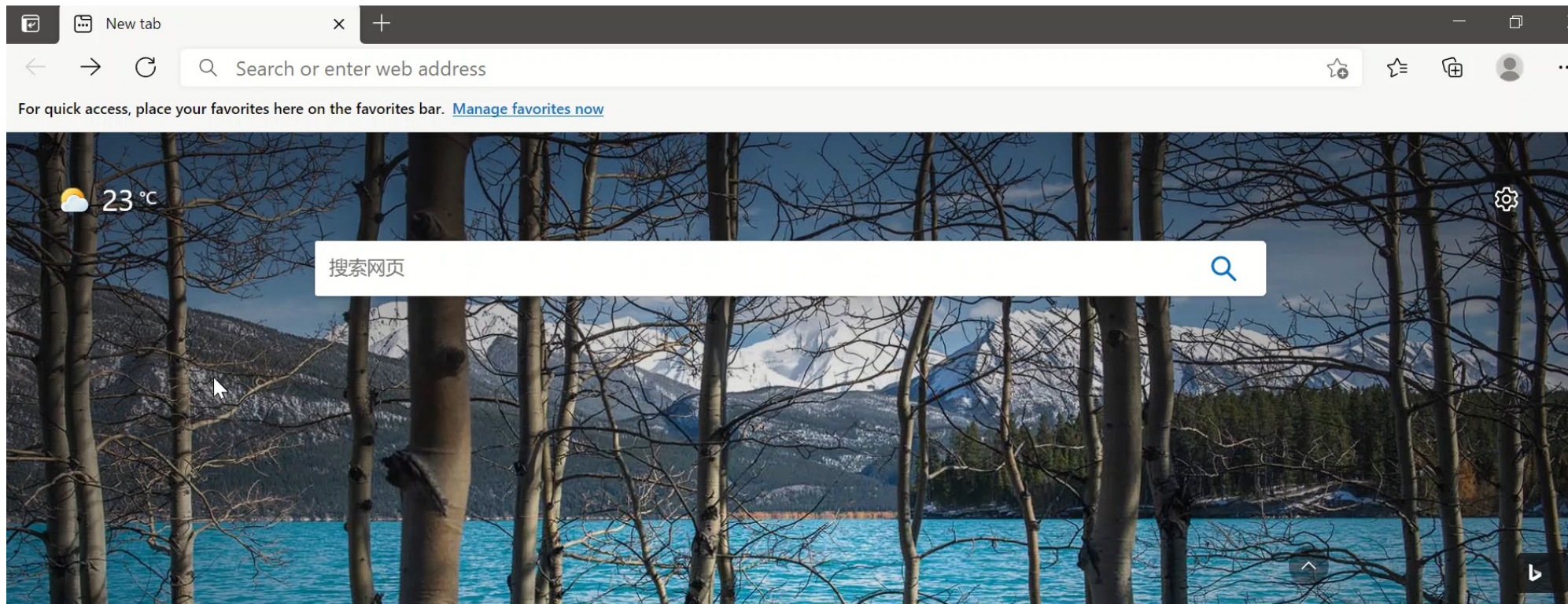


```
1 ('Gutenberg', 7)
2 ('Complete', 3)
3 ('Works', 3)
4 ('William', 64)
5 ('Shakespeare,', 1)
6 ('by', 2791)
7 ('This', 1101)
8 ('eBook', 2)
9 ('for', 5587)
10 ('the', 23104)
11 ('cost', 32)
12 ('and', 18173)
13 ('with', 6701)
14 ('almost', 135)
15 ('You', 1508)
16 ('copy', 14)
17 ('it', 4875)
18 ('or', 1717)
19 ('under', 196)
20 ('terms', 45)
21 ('License', 1)
22 ('included', 1)
23 ('**', 4)
24 ('a', 12472)
```



Visualize Real-time Data with Flask

- We will develop a web app to visualize real-time data
 - A database to store the constantly generated real-time data
 - A HTML chart to visualize and monitor the real-time data





Lab Preparation

- Configure the develop environment locally
 - Create an empty directory *lab6* and go to the directory `cd lab6`
 - Install the virtual environment `pip install virtualenv`
 - Create a virtual environment `virtualenv lab6`
 - Activate the virtual environment
 - `lab6\Scripts\activate` (windows)
 - `Source lab6/bin/activate` (macOS)
 - Install flask `pip install flask`
 - Install mysql-connector `pip install mysql-connector`



Store Data Constantly to Database

- Create a new table in a new database

```
drop database if exists lab6;  
create database if not exists lab6;  
  
use lab6;  
CREATE TABLE IF NOT EXISTS Monitor  
(  
  id int(11) unsigned NOT NULL AUTO_INCREMENT,  
  num int(11) DEFAULT NULL,  
  ctime bigint(11) DEFAULT NULL,  
  PRIMARY KEY (id)  
) ENGINE=InnoDB AUTO_INCREMENT=0 DEFAULT CHARSET=utf8;
```




Store Data Constantly to Database

- Create the *write.py* file to insert data to the database

```
1  import time
2  import mysql.connector
3  import json
4  import random
5
6  def db_connection():
7      mydb = mysql.connector.connect( host = 'comp4442-lab3.cicfnxyayefu.us-east-1.rds.amazonaws.com',
8      user = 'admin',
9      port = '3306',
10     database = 'lab6',
11     passwd = '12345678',
12     autocommit = True)
13     return mydb
14 mydb = db_connection()
15 cur = mydb.cursor()
16
17 def genData():
18     number=15 + 5*random.randint(1,10)
19     Time = int(time.time())
20     data = {}
21     data['time'] = Time
22     data['number'] = number
23     return data
24
25 def execute():
26     data = genData()
27     sql = "insert into Monitor(num,ctime) values ({0},{1})".format(data['number'],data['time'])
28     ret = cur.execute(sql)
29
30 while True:
31     execute()
32     time.sleep(1)
```

➔ Database connection

➔ Generate real-time data

➔ Insert data into database



Build a HTML chart with JavaScript

- Then, we create the HTML chart (Highchart)

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <title>Real-time Monitoring</title>
5   <script src="http://cdn.hcharts.cn/jquery/jquery-1.8.3.min.js"></script>
6   <script src="http://cdn.hcharts.cn/highstock/highstock.js"></script>
7   <script src="http://cdn.hcharts.cn/highcharts/modules/exporting.js"></script>
8 </head>
9
10 <body>
11 <div id="container" style="min-width:400px;height:400px"></div>
12 </body>
13
14 <script type="text/javascript">
15 $(function () {
16   $.getJSON('/data', function (data) {
17     // Create the chart
18     $('#container').highcharts('StockChart', {
19       chart: {
20         events: {
21           load: function () {
22             var chart = $('#container').highcharts();
23             var series = chart.series[0];
24             // 2 seconds interval
25             setInterval(function () {
26               $.getJSON("/data", function (res) {
27                 $.each(res, function (i, v) {
28                   series.addPoint(v);
29                 });
30               });
31             }, 2000);
32           }
33         },
34         rangeSelector: {
35           selected: 1
36         },
37         title: {
38           text: 'Real-time Monitoring'
39         },
40         series: [{
41           name: 'Real-time Number',
42           data: data,
43           tooltip: {
44             valueDecimals: 2
45           }
46         }]
47       },
48     });
49   });
50 });
51 </script>
52 </html>
```



Claim the location of Highchart JavaScript



Set the interval as 2s, asynchronous update



Visual Real-time Data with Flask

- Bind the function `getdata()` to the url `"/data"`

```
1 from flask import Flask, request, render_template
2 import json
3 import mysql.connector
4
5 application = Flask(__name__)
6
7 def db_connection():
8     mydb = mysql.connector.connect( host = 'comp4442-lab3.cicfnxyayefu.us-east-1.rds.amazonaws.com',
9     user = 'admin',
10    port = '3306',
11    database = 'lab6',
12    passwd = '12345678',
13    autocommit = True)
14
15    #print("successfully connect to the database")
16
17    return mydb
18
19 mydb = db_connection()
20 cur = mydb.cursor()
21
22 @application.route("/")
23 def index():
24     return render_template("monitor.html")
25
26 tmp_time = 0
27
28 @application.route("/data")
29 def getdata():
30     global tmp_time
31     if tmp_time > 0 :
32         sql = "select ctime,num from Monitor where ctime >%s" %(tmp_time)
33     else:
34         sql = "select ctime,num from Monitor"
35
36     cur.execute(sql)
37     datas = []
38     for i in cur.fetchall():
39         datas.append([i[0], i[1]])
40
41     if len(datas) > 0 :
42         tmp_time = datas[-1][0]
43
44     return json.dumps(datas)
45
46
47 if __name__ == "__main__":
48     application.run(port=5000,debug=True)
```



Database connection

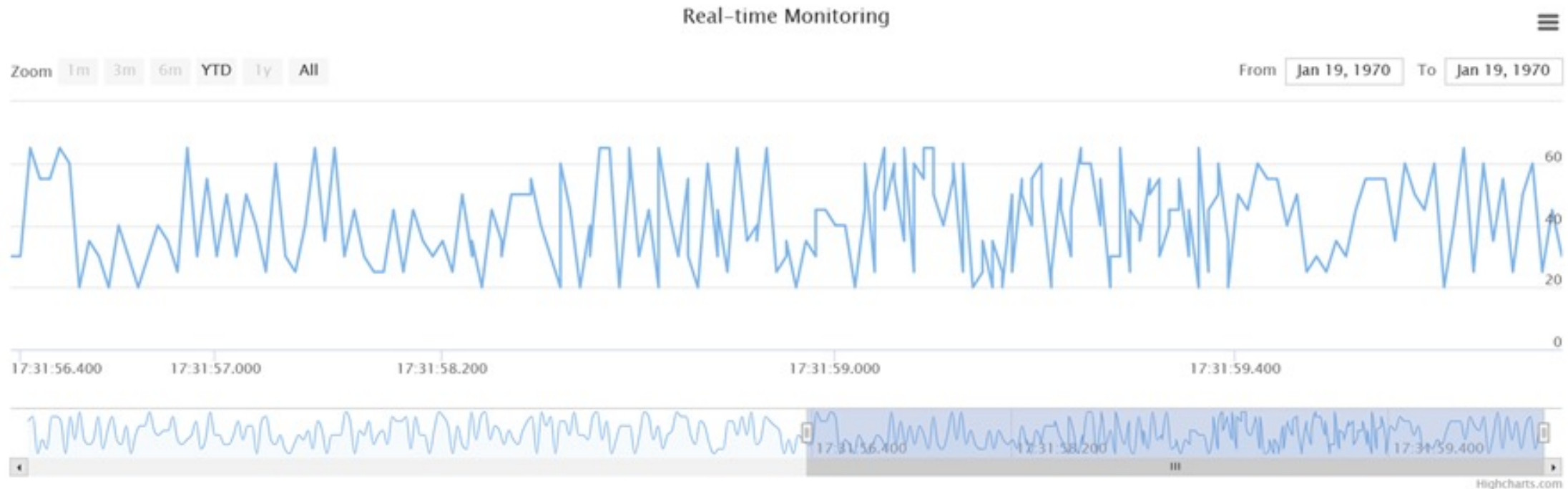


Query the data from the database and
update the front-end webpage



Visual Real-time Data with Flask

- Check the chart to visualize data at “<http://127.0.0.1:5000/>”.





Deploy Code in AWS Beanstalk

- To deploy the code, we need to configure the requirements.txt.
 - `pip freeze > requirement.txt`
- Pack the following files and directories into a ZIP file.

```
requirements.txt
1 click==8.0.3
2 Flask==2.0.2
3 itsdangerous==2.0.1
4 Jinja2==3.0.3
5 MarkupSafe==2.0.1
6 mysql-connector==2.2.9
7 Werkzeug==2.0.3
8
```

Name	Date modified	Type	Size
lab6	4/13/2021 8:06 PM	File folder	
<input checked="" type="checkbox"/> templates	4/13/2021 4:42 PM	File folder	
<input checked="" type="checkbox"/> application.py	4/13/2021 9:06 PM	Python File	2 KB
lab6_project.zip	4/13/2021 9:09 PM	Compressed (zipped)...	3 KB
<input checked="" type="checkbox"/> requirements.txt	4/2/2021 10:05 PM	Text Document	1 KB
test.py	4/13/2021 9:04 PM	Python File	1 KB
write.py	4/13/2021 10:09 PM	Python File	1 KB



Deploy Code in AWS Beanstalk

- **Restore** Beanstalk environment and upload the ZIP file as we do in lab-01 and lab-04.
- Run the write.py locally

The screenshot shows the AWS Elastic Beanstalk console. On the left sidebar, the 'Applications' tab is selected. The main area displays a table of applications. The application 'COMP4442-lab4' is highlighted. The 'Actions' dropdown menu is open, showing options like 'Create environment', 'Delete application', 'View application versions', 'View saved configurations', and 'Restore terminated environment'. The 'Restore terminated environment' option is highlighted with a red box.

Application name	Environments	Date created	Last modified	ARN
COMP4442-lab4	Comp4442lab4-env-1	2021-04-02 22:16:51 UTC+0800	2021-04-02 22:16:51 UTC+0800	arn:aws:elasticbeanstalk:us-east-1:762251576369:applica



Stop Elastic Beanstalk

- Access the beanstalk

The screenshot shows the AWS Management Console interface. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and user information. The main content area is divided into two columns. The left column, titled 'AWS services', contains a 'Recently visited services' section where 'Elastic Beanstalk' is highlighted with a red box. Below this is an 'All services' section with a grid of service categories: Compute (EC2, Lightsail, Lambda, Batch, Elastic Beanstalk, Serverless Application Repository, AWS Outposts, EC2 Image Builder), Containers (Elastic Container Registry, Elastic Container Service, Elastic Kubernetes Service), Storage (S3, EFS), Quantum Technologies (Amazon Braket), Management & Governance (AWS Organizations, CloudWatch, AWS Auto Scaling, CloudFormation, CloudTrail, Config, OpsWorks, Service Catalog, Systems Manager, AWS AppConfig, Trusted Advisor, Control Tower, AWS License Manager, AWS Well-Architected Tool), and Security, Identity, & Compliance (IAM, Resource Access Manager, Cognito, Secrets Manager, GuardDuty, Inspector, Amazon Macie, AWS Single Sign-On, Certificate Manager, Key Management Service, CloudHSM, Directory Service, WAF & Shield, AWS Firewall Manager, Artifact, Security Hub, Detective). The right column contains promotional banners for the AWS Console Mobile App, Amazon Redshift, AWS Fargate, and Amazon S3 backup and restore solutions.



Stop Elastic Beanstalk

- Terminate the environment

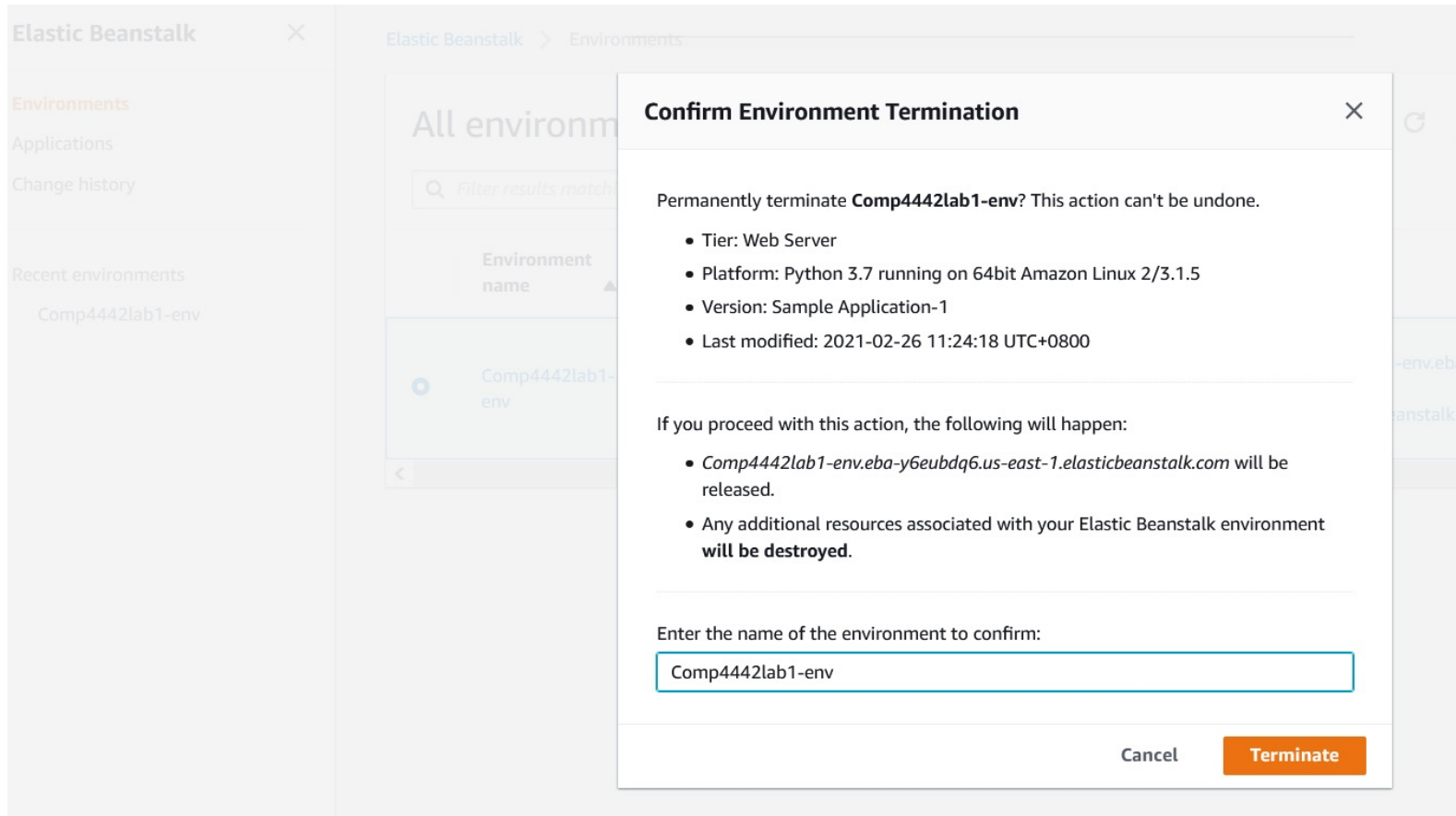
The screenshot shows the AWS Elastic Beanstalk console. The left sidebar has 'Elastic Beanstalk' selected. The main area shows 'All environments' with a table of environments. The first environment, 'Comp4442lab1-env', is highlighted. A red box labeled '1' is around the 'Actions' icon for this environment. A red box labeled '2' is around the 'Actions' dropdown menu. A red box labeled '3' is around the 'Terminate environment' option in the dropdown menu.

Environment name	Health	Application name	Date created	Last modified	URL
Comp4442lab1-env	OK	COMP4442-lab1	2021-02-26 11:12:43 UTC+0800	2021-02-26 11:24:18 UTC+0800	Comp4442lab1-env-y6eubdq6.us-east-1.elasticbeanstalk.com



Stop Elastic Beanstalk

- Confirm your operation





Lab Overview: What to Learn

- Introduction to Amazon Web Services (AWS)
- Data storage and management via AWS S3
 - Learn how to upload, access, and migrate data on AWS
- Create and manage databases via AWS RDS
 - Learn how to make use of database to support complex applications
- Building a dynamic web app via AWS Beanstalk
 - Learn how to use web server to deploy applications
- Big Data Analytics via AWS Spark
 - Learn how to use spark to analyze big data
- Visualize real-time data with dynamic web app



Q&A