CS 378: Modern Web Applications
University of Texas at Austin
Instructor: Devdatta Kulkarni

Deploying web app on AWS EC2 (IaaS approach)

1) Create AWS account by signing up at https://aws.amazon.com/console/
  - You will be required to provide credit card details. You will be charged only if usage of AWS resources goes beyond specified limits for the free tier

2) Create an EC2 instance using Ubuntu 14.04 image.

3) Create a security group for the EC2 instance to allow traffic to port 8080

4) Download the SSH private key to your local machine

5) Get the Public DNS of the EC2 instance and login to it using command of the following form:

```
ssh -i <Private-key-file.pem> ubuntu@<Public-DNS-of-EC2-Instance>
```

6) Once logged in, install required software packages on the instance:

```
- sudo apt-get update
- sudo apt-get upgrade
- sudo apt-get install git maven
- wget http://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.8/bin/apache-tomcat-8.5.8.tar.gz
- gunzip apache-tomcat-8.5.8.tar.gz
- tar -xvf apache-tomcat-8.5.8.tar
- ./apache-tomcat-8.5.8/bin/startup.sh
- sudo apt-add-repository ppa:webupd8team/java
- sudo apt-get update
- sudo apt-get install oracle-java8-installer
```

7) Clone ModernWebApps repo

- https://github.com/devdattakulkarni/ModernWebApps.git

8) Build js-example

```
- cd ModernWebApps/JavaScript-Example/js-example
- mvn clean compile
- mvn package war:war
```

9) Deploy js-example

```
- cp target/js-example.war ~/apache-tomcat-8.5.8/webapps/.
```

10) Access the application at:

```
http://<Public-IP-Address-of-EC2-Instance>/js-example/ajaxexample.html
http://<Public-IP-Address-of-EC2-Instance>/js-example/test.html
```

Deploying web app on Heroku (PaaS approach)

1. Signup for a Heroku Account
- https://www.heroku.com/

2. Install Heroku CLI
- https://devcenter.heroku.com/articles/heroku-command-line

3. Copy Procfile, pom.xml, system.properties, and ajaxexample.html from Heroku folder inside JavaScript-Example to js-example/
- Copy Procfile, pom.xml, and system.properties in js-example/
- Copy ajaxexample.html in js-example/

4. Copy js-example to a temporary folder
- cp js-example /tmp/.

5. Switch to the js-example folder in the temporary location
- cd /tmp/js-example

6. Setup up application on Heroku
- heroku create <unique-app-name>

7. Initialize the application with source code
- git init
- git add .
- git commit -m “Deploying app to Heroku”

8. Deploy to Heroku
- git push heroku master

9. Test the application
- https://<unique-app-name>.herokuapp.com/test.html
- https://<unique-app-name>.herokuapp.com/ajaxexample.html

References:
- https://devcenter.heroku.com/articles/deploy-a-java-web-application-that-launches-with-jetty-runner

Deploying web app on AWS Elastic Beanstalk (PaaS approach)

1) Create AWS account by signing up at https://aws.amazon.com/console/
- You will be required to provide credit card details. You will be charged only if usage of AWS resources goes beyond specified limits for the free tier

2) Go to the folder of js-example

3) Create war file
- mvn package war:war

4) Copy the war file to ROOT.war
- cd target
- cp js-example.war ROOT.war

5) Go to the ElasticBeanstalk console

6) Create application
- Choose Tomcat as the application platform
- Choose the option to upload ROOT.war
- Choose the default options for the rest of the options
- Deploy the application

7) Once the application is READY, you will see the application box on the web console becomes green.

8) Access the application url and the html files at:
http://<app-name>-<region>.elasticbeanstalk.com/test.html
http://<app-name>-<region>.elasticbeanstalk.com/ajaxexample.html

References:

http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/Welcome.html
http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create_deploy_Java.html
http://beanstalker.ingenieux.com.br/beanstalk-maven-plugin/usage.html

Deploying web app on Google App Engine (PaaS approach)

References:
https://cloud.google.com/appengine/docs/java/tools/maven