Home Assignment 2

Overview

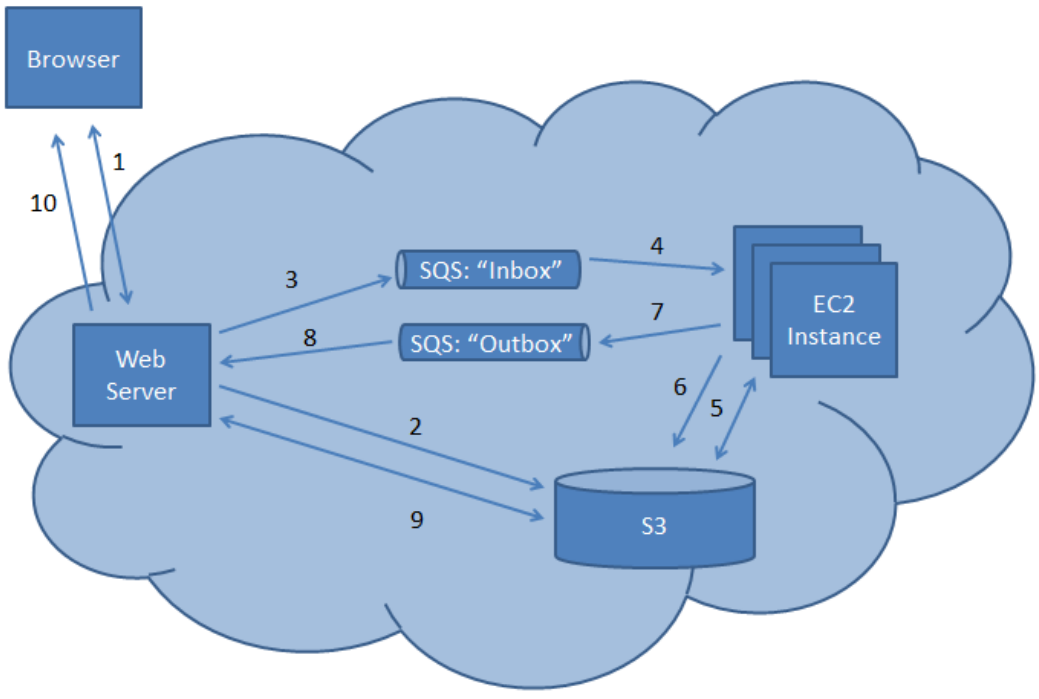
In this assignment you will build further upon home assignment 1. Home assignment 1 was, as you recall, the back-end of a scalable web application using Amazon AWS resources. Moreover since your first assignment targeted Infrastructure-as-a-Service (IaaS) this home assignment will target Platform-as-a-Service (PaaS).

The goal of this assignment is to make you build the front-end of the web application you started in the previous assignment. The PaaS you will use is the Amazon Beanstalk.

Once again there are various ways to do this homework, Beanstalk supports Java, .NET, PHP, Python, Ruby, and Docker. However, I will only give details on how to do it using Eclipse, Java (.jsp), and Apache Tomcat. But please, feel free to do it in whatever way you wish, as long as you meet the requirements.

Application Structure

The structure of this assignment will build on the one from home assignment 1 and can be seen below. As you can see the only difference is that the client-app, which previously was outside the cloud-structure, has been replaced by a web-server that now is within the cloud structure. This allows your application as a whole to be able to scale better. The good thing with a PaaS is that the provider will do the scaling and load balancing for you.



Requirements

The requirements for the web-server is as follows:

- The web server should be hosted using Amazon Beanstalk
- The functionality should be the same as for the Client-Application in home assignment 1:
  - User inputs a comma-separated list of numbers
  - Puts this list in the S3-bucket (and remembers the key/pointer to this object).
  - Puts a message in the SQS-inbox with a key/pointer to the object in the S3-bucket, along with the process to be executed on these numbers.
  - Waits until a response is generated in the SQS-outbox (which should contain a pointer to the new, and processed, object in the S3-bucket along with the process executed).
  - Reads the result from the S3-bucket
  - Prints the results along with the original numbers, and the process that was executed.
  - Deletes the result-message from the SQS-outbox.

Again the available processes should be:

- Min
- Max
- Product
- Sum
- Sum-of-squares

Remember that you might now serve several users simultaneously, and therefore it will be extra important that you figured out a clever way to relate the correct user-request with the correct response in the previous homework. Also, there is no need for a fancy GUI here, some good ole' fashioned text boxes will do fine..

Creating the web server

To get you started with AWS Beanstalk, I recommend following this walkthrough on how to create, view, deploy, and update your application.

When you know a little more about how to work with the AWS Beanstalk I suggest that you plow through this guide on how the AWS Beanstalk work. It contains some overview of the architecture structure, along with some design considerations.

Using Eclipse to create a Java web application

This tutorial takes you through the process of creating a new Java web application in Eclipse, test it locally (using Apache Tomcat), and then deploy it on AWS Beanstalk. I suggest that you work your way through it since it will give you some good sample code and make it easier for you to do the assignment :)

If you need to install Apache Tomcat on your computer I found this site very useful!

If you have no experience on working with `java.jsp` (like me), take a look at this very basic tutorial (it will teach you all you need to know to do the assignment).

Some useful links

- Article on creating a similar application (By Amazon)
- AWS Java API
- Web Application Hosting in the AWS Cloud (Amazon White Paper)
- General AWS Documentation
- Some more articles and tutorials
- AWS Java Developer Center