Note on VPC configuration for Assignment 2 (step 3)

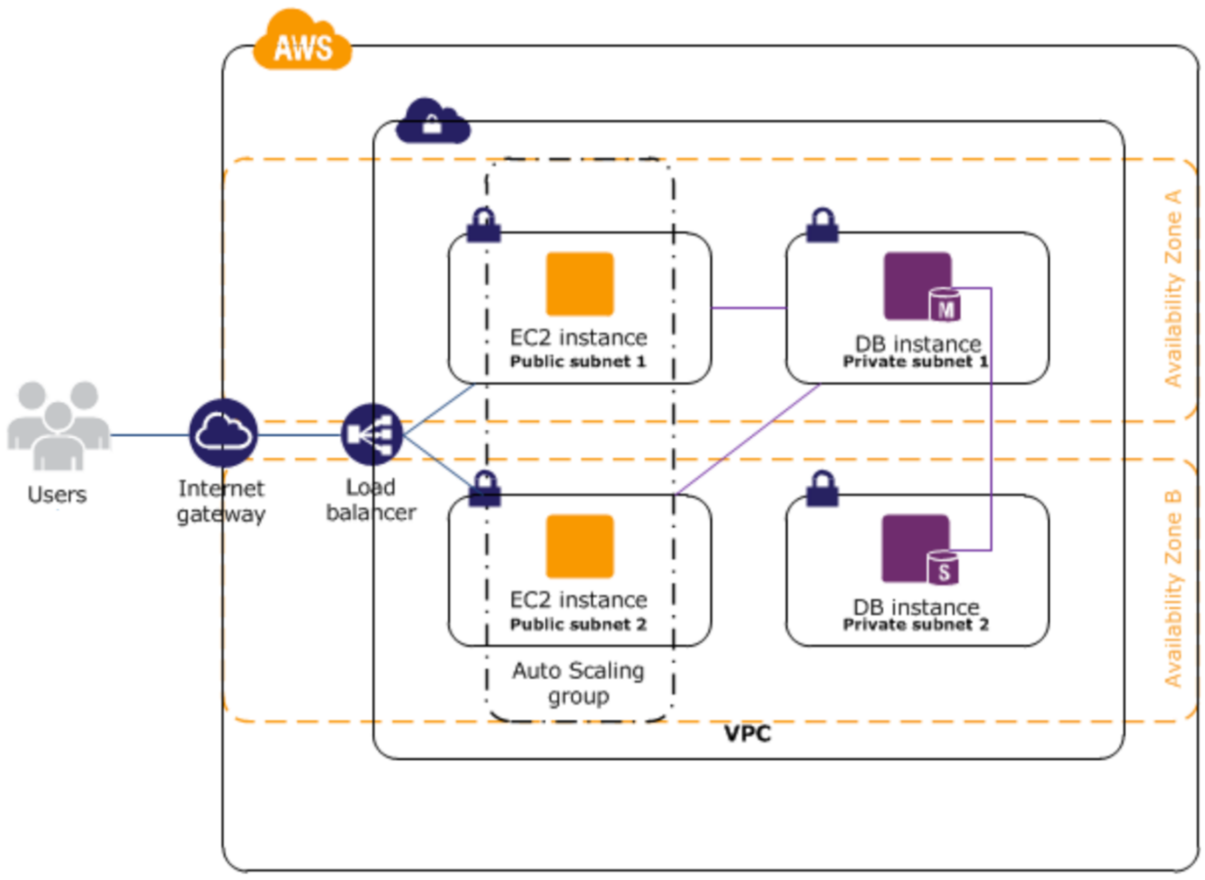
Step 3 of Assignment 2 specifies:

"Creation of a VPC with public and private subnets into which your application will be deployed. Creation of suitable security groups."

The following is a clarification of exactly what is required, especially if you choose not to deploy a database on a private instance as part of your assignment.

- You are required to create a VPC.
- This VPC should have public and private subnets in at least two different availability zones.
- You web server instances should run in this VPC (in public subnets).
- If you deploy a database instance, it should be in a private subnet. We don't explicitly require you to deploy a - database instance; this is "additional functionality". You can use a public database service or have no database at all.
- Security groups are per-VPC. For those without a database instance, "creation of suitable security groups" just means security groups for your load balancer and public instances.

The picture below shows a typical web application architecture.



Note on Assignment 2 (steps 9 and 10)

Steps 9 and 10 from the Assignment specification state :

9. Generation of test traffic to the load balancer – e.g. using curl/wget or a web testing tool.
10. Show that the load is distributed across more than one web server – e.g. by viewing web server or other logs. Screenshots and a brief explanation in your report will suffice for this.

For step 10 of Assignment 2, you are required to show that the load is distributed across more than one web server - e.g. by viewing web server or other logs. The **tail** command with the **-f** option is handy for reading "live" logs.

This link should be of benefit in understanding the operation of the ELB and how you can go about testing it. This link should be useful in deciding which Cloudwatch metrics to use when monitoring your ELB .

To help you generate test traffic to your load balancer in order to test your load balancer you could use a tool such as curl to send a sequence of requests to it, each with a slightly different URL and have a look the server logs afterwards. For example this command would send 100 requests to http://lbdnsname/1/, http://lbdnsname/2/, etc.

```
curl -s "http://lbdnsname/?[1-100]"
```

Look at the logs on each of the target instances (e.g. use the **tail -f** command) and you should see how these requests are distributed. For example the apache access log is available at /var/log/httpd/access_log