**Assignment: 1**

Application patch is released by Engineering team to resolve a bug. However, after applying a patch, another issue which was fixed earlier opens up again. What Engineering could have done wrong in their development and release cycle which caused this issue? Also, what is the resolution for the each case that you would mention as issue?

Answer:

It seems that the developer has created a hotfix branch to resolve the issue, but removed the old code and merge it directly in to master branch without pushing the code to QA branch for testing

Or

It could be due to merge conflict the developer has updated the new code which re-open the old bug again.

To resolve the issue, we need to follow the proper git flow, even if we created a hotfix branch to resolve the issue and we should send a merge request to manager/other team members to review the code and merge. After that the code has to be pushed to QA or UAT environment for testing and then merge it to prod/master branch.

#######################################################

**Assignment: 2**

Assume a tech-stack as J2EE Application Server (J boss, Web logic, Web sphere, etc) Apache as Web Layer and Oracle RAC as DB layer.

J Boss and Apache are working in 3 node cluster. (If not J boss than consider any other application server)

We need to upgrade the application which is deployed on J Boss in HA mode with zero downtime to end customer.

Tell us the methodology on how this can be achieved. Also, let us know different scenarios which need to be taken care for failure scenarios.

Answer:

1) For this we can you the **blue-green deployment** methodology i.e. parallel to the prod(Blue Environment) instance we have to create new environment(Green environment).

We need to test the new environment and once all done, we need to kill the old session and pointed it to old(Blue) environment. In case anything went wrong, we can point the sessions to the Blue environment.

2) Further, If you are using Apache Tomcat, they have the concept called parallel deployment.

3) Also, In AWS, we can achieve this by using ELB's and Route53 using Route53 weighted feature i.e

We will have two same stack (blue and green), and In Route53 we can adjust the weighted to 10% traffic goes to Green(new) environment and 90% traffic goes to blue(old) environment.

With the passage of time, we can change the weight to 50-50% in both blue and green environment.

And further, if all goes well we point 100% traffic to Green environment and discard the blue environment.

Also, in case of failure, our old Blue environment is with us.