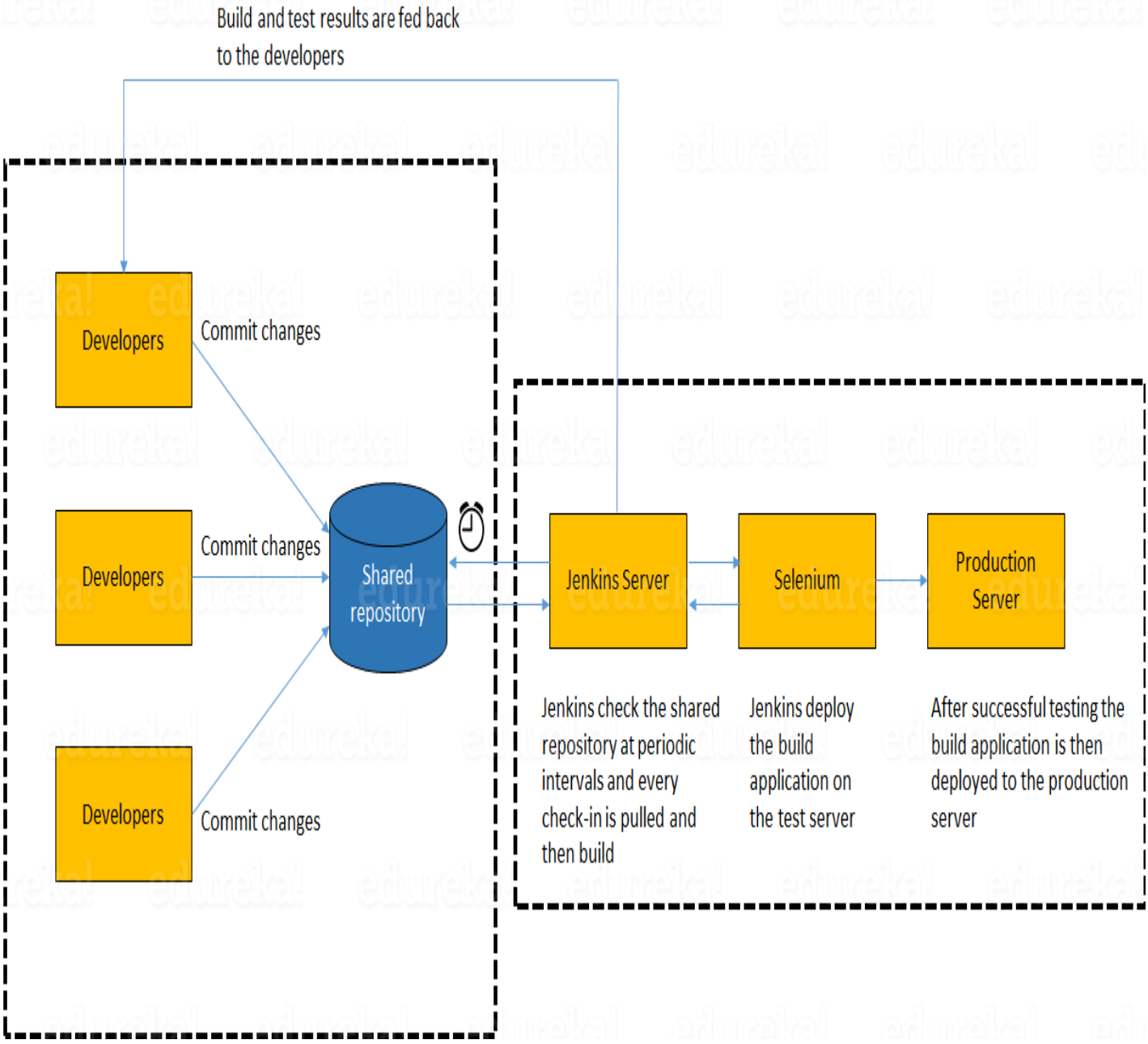
Continuous Integration questions

Now, let's look at Continuous Integration interview questions:

Q1. What is meant by Continuous Integration?

I will advise you to begin this answer by giving a small definition of Continuous Integration (CI). It is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.
I suggest that you explain how you have implemented it in your previous job. You can refer the below given example:



In the diagram shown above:

1. Developers check out code into their private workspaces.
2. When they are done with it they commit the changes to the shared repository (Version Control Repository).
3. The CI server monitors the repository and checks out changes when they occur.
4. The CI server then pulls these changes and builds the system and also runs unit and integration tests.
5. The CI server will now inform the team of the successful build.
6. If the build or tests fails, the CI server will alert the team.
7. The team will try to fix the issue at the earliest opportunity.
8. This process keeps on repeating.

Q2. Why do you need a Continuous Integration of Dev & Testing?

For this answer, you should focus on the need of Continuous Integration. My suggestion would be to mention the below explanation in your answer:
Continuous Integration of Dev and Testing improves the quality of software, and reduces the time taken to deliver it, by replacing the traditional practice of testing after completing all development. It allows Dev team to easily detect and locate problems early because developers need to integrate code into a shared repository several times a day (more frequently). Each check-in is then automatically tested.

Q3. What are the success factors for Continuous Integration?

Here you have to mention the requirements for Continuous Integration. You could include the following points in your answer:

- Maintain a code repository
- Automate the build
- Make the build self-testing
- Everyone commits to the baseline every day
- Every commit (to baseline) should be built
- Keep the build fast
- Test in a clone of the production environment
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment

Q4. Explain how you can move or copy Jenkins from one server to another?

I will approach this task by copying the jobs directory from the old server to the new one. There are multiple ways to do that; I have mentioned them below:
You can:

- Move a job from one installation of Jenkins to another by simply copying the

corresponding job directory.

- Make a copy of an existing job by making a clone of a job directory by a different name.
- Rename an existing job by renaming a directory. Note that if you change a job name you will need to change any other job that tries to call the renamed job.

Q5. Explain how can create a backup and copy files in Jenkins?

Answer to this question is really direct. To create a backup, all you need to do is to periodically back up your JENKINS_HOME directory. This contains all of your build jobs configurations, your slave node configurations, and your build history. To create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

Q6. Explain how you can setup Jenkins job?

My approach to this answer will be to first mention how to create Jenkins job. Go to Jenkins top page, select “New Job”, then choose “Build a free-style software project”. Then you can tell the elements of this freestyle job:

- Optional SCM, such as CVS or Subversion where your source code resides.
- Optional triggers to control when Jenkins will perform builds.
- Some sort of build script that performs the build (ant, maven, shell script, batch file, etc.) where the real work happens.
- Optional steps to collect information out of the build, such as archiving the artifacts and/or recording javadoc and test results.
- Optional steps to notify other people/systems with the build result, such as sending e-mails, IMs, updating issue tracker, etc..

Q7. Mention some of the useful plugins in Jenkins.

Below, I have mentioned some important Plugins:

- Maven 2 project
- Amazon EC2
- HTML publisher
- Copy artifact
- Join
- Green Balls

These Plugins, I feel are the most useful plugins. If you want to include any other Plugin that is not mentioned above, you can add them as well. But, make sure you first mention the above stated plugins and then add your own.

Q8. How will you secure Jenkins?

The way I secure Jenkins is mentioned below. If you have any other way of doing it, please mention it in the comments section below:

- Ensure global security is on.

- Ensure that Jenkins is integrated with my company’s user directory with appropriate plugin.
- Ensure that matrix/Project matrix is enabled to fine tune access.
- Automate the process of setting rights/privileges in Jenkins with custom version controlled script.
- Limit physical access to Jenkins data/folders.
- Periodically run security audits on same.