Jenkins

**Continuous Integration:-**

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly.

* There are three things that determines your code is good.
  + Build
  + Deployment
  + Testing
* Integration errors are two types
  + Compile time errors
  + Runtime errors

**Advantages of Continuous Integration:-**

* The more you integrate frequently the less you need to spend to resolve integration errors.
* It speeds up the development process.

**Tools to do Continuous Integration:-**

* Cruise control
* Jenkins
* Bambo
* Build forge

**Jenkins is like a website that access by using browser**

**Needs to run Jenkins :-**

* Install tomcat
* Deploy jenkins.war in tomcat
* Restart tomcat

**Installation of Tomcat:-**

* First install jdk
* Download tomcat zip file and extract
* Goto **<Tomcat\_Home>/bin**
* Run **startup.sh** script (# ./startup.sh)
* To know tomcat is running or no , **# ps -ef | grep java**

**Installation of Jenkins:-**

* Download **jenkins.war** from internet.
* Copy the war file to **<Tomcat\_home>/webapps**  directory
* Restart tomcat (**run shutdown.sh**)

**Access jenkins application:-**

* In your browser type **http://<tomcat\_ip>:<tomcat\_port\_number>/jenkins**
* **username→ Admin**
* **password→ open /home/username/.jenkins/secrets/initialAdminPassword**
* Manage jenkins is the administrative interface, if you not see manage jenkins in the jenkins home page, that means you are not jenkins administrator
* After install jenkins you have to configure authentication and authorization
  + Go to manage jenkins→ configure global security
  + Tick enable security

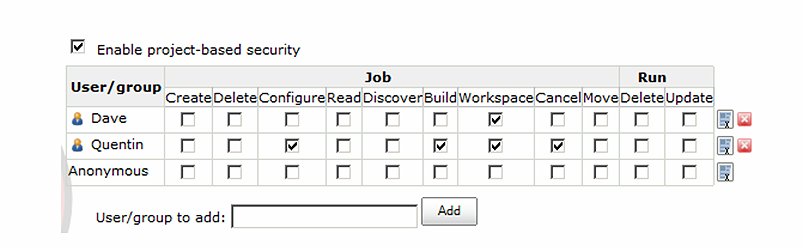
**Configure Authentication in jenkins:-**

* Go to manage jenkins→ configure global security→ security realm
* Select one of the options, you want to configure security using **Active directory or LDAP** , you need to install separate plugins.

****

**Configure Authorization in jenkins:-**

* Go to manage jenkins→ configure global security→Authorization
* Select the options, you want to configure authorization plugins, install **Role-based-Authorization strategy or matrix authorization strategy**
* Example , select matrix authorization strategy and add users and groups give permissions.



**Plugins in Jenkins:-**

* Plugins are used to extend the functionality of jenkins.

**Install plugins in jenkins:-**

* Go to manage jenkins→ manage plugins
* You see four tabs like
  + Updates
  + Available
  + Installed
  + Advanced
* Go to available , search and install the plugin what you want to install.

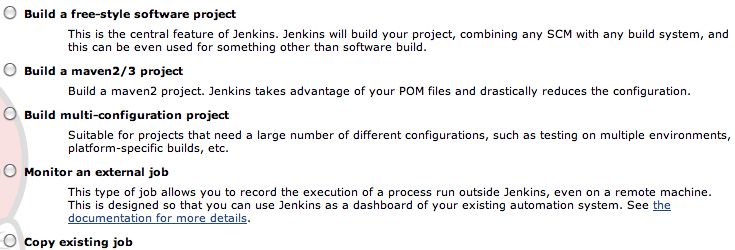
**Install plugins manually:-**

* Download plugin related .hpi file
* Goto manage jenkins→ manage plugins→ advances → upload plugin
* In choose file browse the downloaded .hpi file and upload.

**Common plugins that are used in jenkins:-**

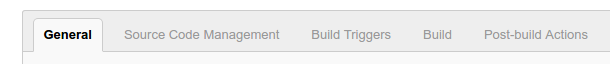
* Active directory
* Matrix authorization strategy
* Audit trail plugin
* Build metrics
* Credentials plugin
* Deploy plugin
* Deploy to websphere container
* Downstream build view
* Html publisher plugin
* Jenkins disk usage plugin
* Jenkins environment file plugin
* Jenkins ssh plugin
* Jenkins subversion tagging plugin
* Jenkins text finder plugin
* Publish over ssh
* Retry failed builds
* Warning plugin
* Workspace cleanup plugin

**Different jenkins job types:-**

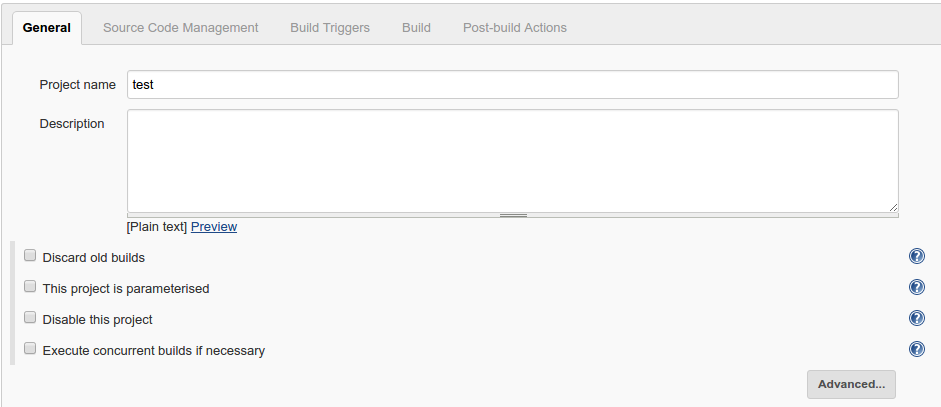
* Free style project (most common project)
* multi -configuration project
* Maven ⅔ project
* Monitor external job
* Copy existing job(used to copy existed job)

**Jenkins job categories:-**

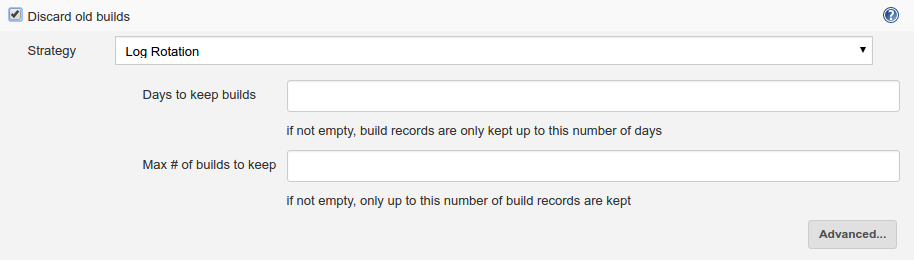
* General category
* Source Code management(SCM)
* Build triggers
* Build
* Post-build Actions



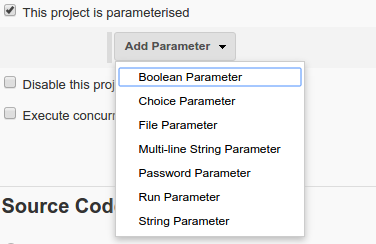
**General category:-**



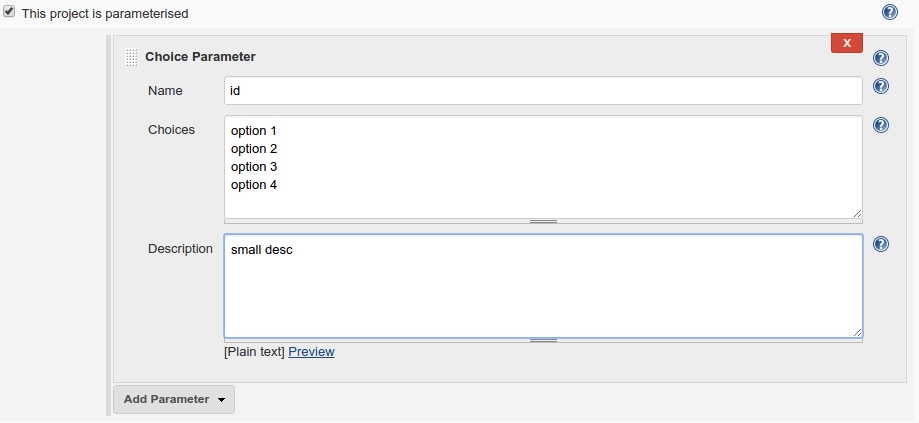
* **Discard old builds:-**

****

* **This project is parameterised:-** 
  + Used to pass the parameters at runtime
  + You want to see more options then you need to install plugins



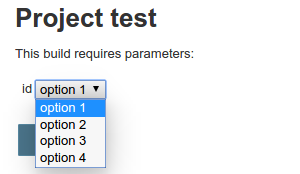
EX:-**choice parameter**

****

* If you do with parameters you get **build with parameters** in item main page

Screenshot from 2016-08-11 12-22-29.png

* After click on build with parameters option.



* **Disable this project:-**

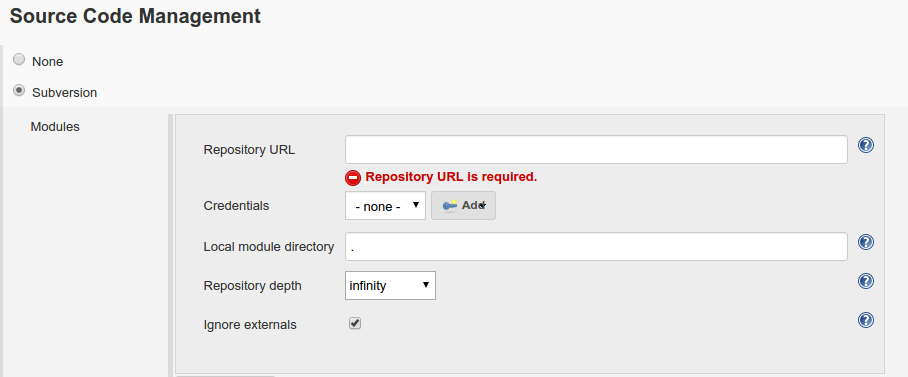
If you tick this option you will not able to run the job.

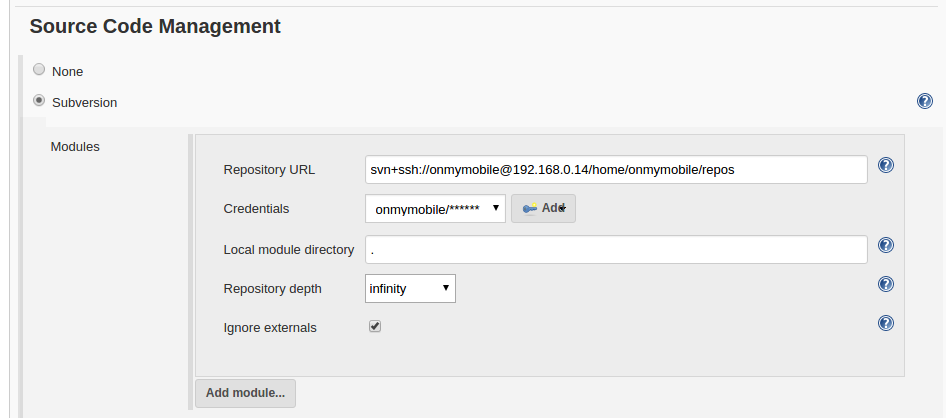
* **Execute concurrent builds if necessary:-**

Used to execute builds concurrently

**Source code management category:-**

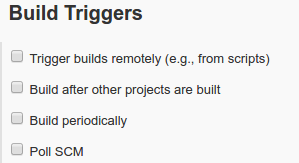
* It is used to check out the repositories from url
* You need to install plugins for example **SVN**

****

****

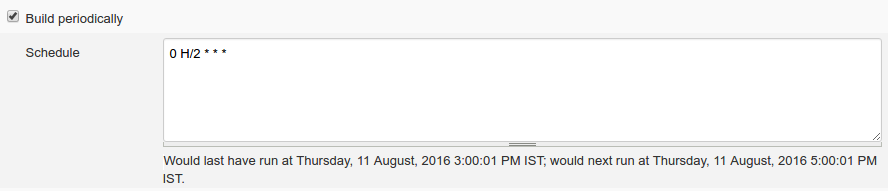
**Build triggers category:-**

Build triggers will have 4 options



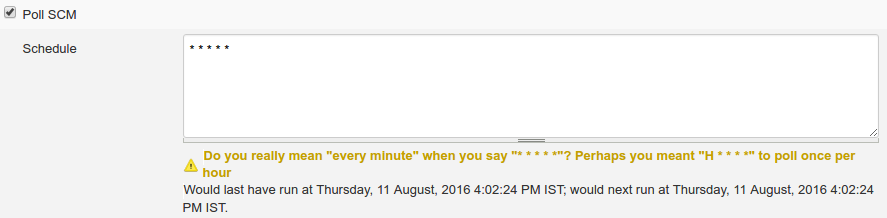
**Build periodically:-**

It is used to schedule a build periodically



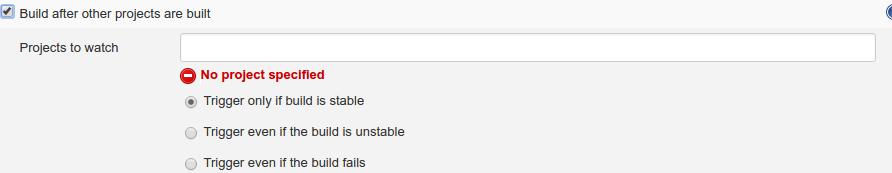
**Pool SCM:-**

If you want to build for every checkin you use poll scm



**Build after other projects are built:-**

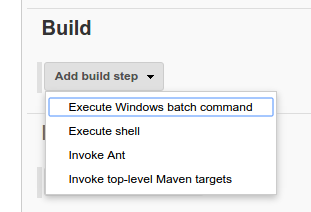
* used to specify dependence between jobs

****

**Jenkins version will see bottom right of every page.**

**Build category:-**

* Actual work of jenkins will be done under build category.
* Build supports 4 types:-
  + Execute windows batch commands
  + Execute shell
  + Invoke top level maven target
  + Invoke Ant (Need to install plugin)



**Execute windows batch command:-**

* Used to execute commands in windows box.

**Execute shell:-**

* Used to execute linux commands.

**Invoke top-level Maven targets:-**

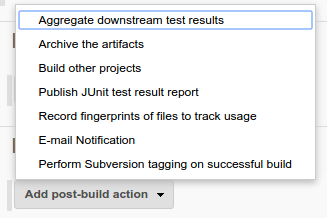
* Used to build , using maven.

**Invoke Ant:-**

* Used to build using Ant.

**Post-build Actions:-**

* After jenkins job is completed, you want to execute any other tasks, you use **post-build actions**
* Like sending emails notifications when build fails etc.



**Jenkins home directory(.jenkins):-**

* Jenkins have a home directory i.e, $HOME/.jenkins
* Every job have a work space inside

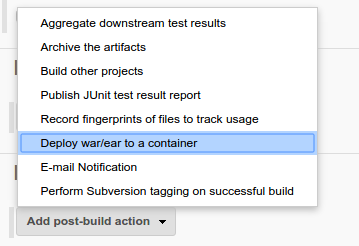
**<JENKINS\_HOME>/jobs/<JOB\_NAME>/workspace**

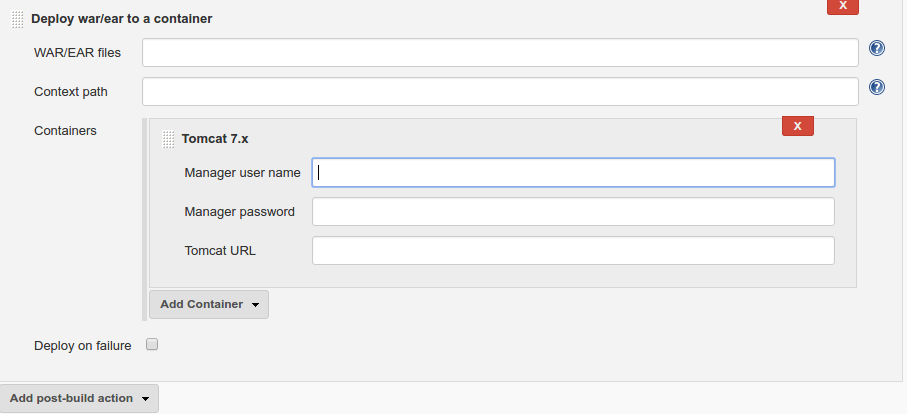
**Console output:-**

* Suppose your build is failed , by using console output logs you may find the bugs and troubleshoot them.

**Deployment:-**

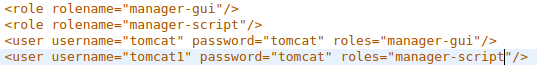
* Deployment will be done in two ways
  + Using plugins (**need to install deploy to container plugin**)
  + Using shell scripts
* **Deployment Using Plugins:-**
  + To install **Deploy to Container** plugin , goto **manage jenkins→ manage plugins→ available,**  and search for Deploy to container plugin.
  + After installing you will see some changes in **post-build actions**

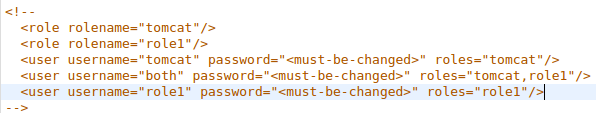




* To activate tomcat manager console, you have to change some configuration change in destination tomcat server.
* NOTE:-you should not activate tomcat manager in production(**live**)

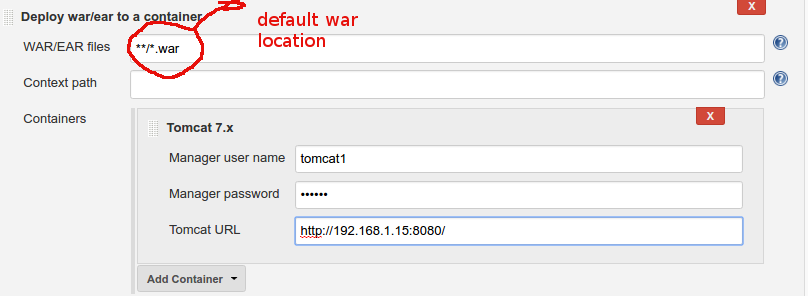
**Steps to activate Tomcat manager console:-**

* Open **<TOMCAT\_HOME>/conf/tomcat-users.xml**
* Configure inside of **</tomcat-users>** tag, by default it is in comments, uncomment



* Change to

* Use **manager-script** related username and password in
* After you configure the tomcat manager app, go to jenkins→ post-build actions



* **Deployment Using Shell Script:-**
  + First and pre-requirement is, when you connect to any remote system using jenkins it should not ask any passwords.
  + Without asking passwords you need to generate **ssh public and private keys**
    - #ssh-keygen rsa
    - The keys are stored in **$HOME/.ssh**
    - Copy the public key to remote machines **$HOME/.ssh/authorized\_keys file**
  + **Deploying in tomcat server using shell script:-(in client’s server)**
    - Stop tomcat
      * **ssh user@IP "cd /home/user/apache-tomcat-7.0.70/bin/; ./shutdown.sh"**
    - Take a backup of existing **war** file in **webapps**
      * **ssh user@IP "cp /home/user/apache-tomcat-7.0.70/webapps/helloWorld-0.1-dev.war /home/user/apache-tomcat-7.0.70/bkp/HelloWorld.2508.war"**
    - Remove the **war** file present inside **webapps**
      * **ssh user@IP "rm-rf /home/user/apache-tomcat-7.0.70/webapps/helloWorld-0.1-dev\*"**
    - Copy the new **war** file from **jenkins** to **tomcat→ webapps directory**
      * **scp /home/user/war/helloWorld-0.1-dev.war user@IP:/home/user/apache-tomcat-7.0.70/webapps**
    - Start tomcat
      * **ssh user@IP "cd /home/user/apache-tomcat-7.0.70/bin; ./startup.sh"**

**Jenkins home directory:-**

* Goto **manage jenkins→ configure system,** home directory appears
  + Screenshot from 2016-08-26 13-32-27.png

**# of executer:-**

* It configure the no.of jenkins jobs , that are run at a time.

**Q:How do you change the Home directory in Jenkins:-**

* **Ans:-**set an environmental variable JENKINS\_HOME=”/opt/jenkins” in etc/profile

**Q:How do you restart jenkins?**

* **Ans:-** in browser type **http://<tomcat\_ip>/jenkins/restart**

**Or**

**Restart tomcat**

**Email settings in jenkins:-**

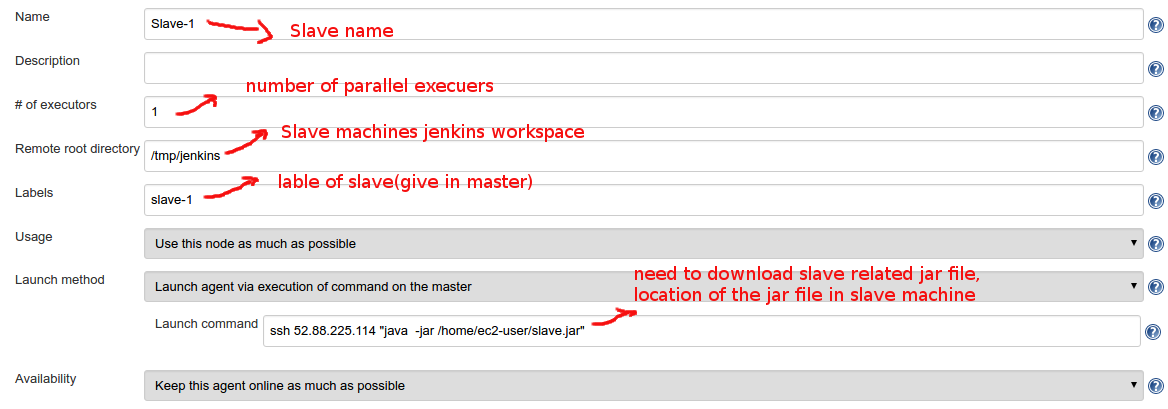
* Goto manage jenkins → configuration→ email notifications
* And give SMTP server details(Simple Mail Transfer Protocol)

**Q:How do you configure Ant,Maven, jdk in jenkins?**

* **Ans:-** go to manage jenkins → configuration→global properties→ tools location

**Distributed builds:-**

* If there is lot of jenkins jobs in your project, the jenkins master is overloaded to run those jobs, thats why we distribute some jobs to another slave machine.
* Some jobs in master and some in slave
* **Slave configuration:-**
  + Go to Manage jenkins→ manage node→ new node



* To configure slave in your job, you need to install **node and label parameter plugin.**
* After the plugin installation completed you will see new option in **projects general category**

****

* Tick the **restrict where this project can be run** and see the option **label expression**, here give the label of slave machine.
* Slave configuration is completed
* This is one time activity

**Deployments:-**

Deployments involves three things

* Code deployments
* Database deployments
* Configuration changes deployment

**Code deployments:-**

* Deploying a WAR and EAR to an application server is called Code deployments

**Database deployments:-**

* Executing our SQL files to the DB server
* There are many tools that are used to database deployments
  + EX:- flyway tool
* There are some plugins in jenkins that are used to database deployments
  + EX:- sql-ci

**Configuration changes deployments:-**

* To change the configurations in jenkins

**Environments:-**

* Environment is a combination of hardware and software, that makes your application work.(what all are makes your website run)
* Here , environment is nothing but website, hardware is the server and software is the code.
* There are different types of Environments
  + Dev Environment
  + QA Environment
  + Performance Environment
  + UAT Environment
  + Production Environment
* **Dev Environment:-**
  + This environment is for Developers
  + The code in Dev Environment can stable and unstable
  + Developers own this environment
  + Developers will perform **Unit Testing** in this environment
  + Developers certify the code in this this environment
    - dev.facebook.com
  + Developments are very frequent in this environment
  + Daily four times in a day deployments happens in their environment
* **QA Environment:-**
  + This environment for QA team to perform **Functional Testing**
  + Builds and deployments are performed by **Build and Release Team**
  + This environment is supposed to be stable code only
  + Deployments to this environment happens twice a week
  + Builds and deployments to this environment should be approved by QA manager
  + Once QA manager certifies this code to this environment , it will be deployed to Performance Environment
* **Performance Environment:-**
  + This environment is for Performance QA to perform **Load Testing.**
  + Deployments to this environment happens three to four times per Release
  + Performance environment is exact replica of production environment in terms of Hardware
* **UAT Environment:-(User Acceptance Testing)**
  + This environment is for client QA to perform Testing
  + This is the final stage before production
  + Deployments happens two to three times for release
* **Production Environment:-**
  + This is the live environment, where end client will use this application

**Q:-How much time take to build an application in your company?**

**Ans:- Build**

* My application have 6 modules
* Every module will have one WAR as output
* Every module will have one jenkins job for build, so we have 6 jenkins jobs for build
* And we have 6 war files from our build process
* And to build each war file it takes 5 to 7 mins , so to build 6 war files it takes approximately 30 to 45 mins (Assume it is done one by one)

**Deployments:-**

* Code Deployments: 16-20 mins
* DB Deployments: 15-20 mins
* Configuration changes: 10-15 mins
* Approximately for Deployment 45-50 mins

**How many servers in your project have?**

**Ans:-**

* 6 servers for code changes (6 modules)
* 1 database server
* In production there are 40 to 50 servers

**Release notes:-**

* Release notes contains all the information that is needed for you to do a successful build and deployments.
* Release notes is prepared by development team.

**Contents of Release notes:-**

1. Introduction
   1. Changes to the release notes
   2. Scope
2. **New infrastructure and system requirement**
   1. **New hardware requirement**
   2. **New partner connectivity / changes**
   3. **Changes in service deployment location**
   4. **Ip white listings and vpn setups**
   5. **Certificate changes**
   6. **New software installation**
   7. **Software version upgrade**
3. New features / functional changes
4. Resolved issues
   1. Included UAT defects
   2. Include production incident fixes
   3. Included internal QA defects fixes
   4. Included platform and tools improvement
5. Dropped features functional changes
6. Known issues , limitations and restrictions
   1. Open defects
   2. Limitations and risk assessments
   3. Restrictions
   4. Exceptions (to above limitations or restrictions)
7. **SVN tag number for Revision**
8. **Change details**
   1. **Application changes**
   2. **Database changes**
   3. **Property value changes**
   4. **Batch job / cron job changes**
   5. **Mail template changes**
9. **Any additional deployment instructions**
10. Related documents

**Svn tag number for revision:-**

|  |  |  |  |
| --- | --- | --- | --- |
| **Service name** | **Repository location** | **Build 13** | **Build 14** |
| card\_services | svn+ssh://user@ip/home/svnroot/logic | 23058 | 23058 |
| card\_db | svn+ssh://user@ip/home/svnroot/db | 22564 | 22564 |

**Database changes:-**

|  |  |  |  |
| --- | --- | --- | --- |
| **Service name** | **Svn main location** | **Change details** | **Build number** |
| Rule engine | svn+ssh://user@ip/home/svnroot/db/cr\_ruleengine.sql | Poc: developer name | 3 |

**Property value changes(config changes):-**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Service** | **File name** | **File location** | **new/**  **change/**  **remove** | **Property name** | **Property value** | **Build number** |
| cashier | c3po.property | config | new | C3po.more statements per connection | 300 | 1 |

**Build plan:-**

|  |  |
| --- | --- |
| **Date** | **Task** |
| 4/17 Tue 10:00 am | Code complete |
| 4/17 Tue 10:00 am to 10:30 am | Prepare release notes |
| 4/17 Tue 10:30 am to 11:30 am | Build the code for R37 build-1 |
| 4/17 Tue 11:30 am to 12:30 pm | Deploy R37 code to QA environment |
| 4/17 Tue All afternoon | Smoke test R37 build-1 |
| 4/18 wed whole day | Smoke test R37 build-1 |
| 4/19 thu 10:00 am | Make any critical bug fix and update release notes |
| 4/19 thu 10:30 to 12:00 | Build and deploy the code to QA environment |
| 4/19 thu all afternoon | Smoke testing in QA environment |
| 4/23 mon | Deploy to perf environment |

**What the build plan follow in your company?**

* All the build and deployments are already planned by build manager, and send build plan, follow build plan.