Project Creation Date	28-march-2024
Documentation Date	13-September-2024
Version Number	1.0.0
Product name	Package-Typing
Team lead	Nadimpalli Pradeep Kumar
Current AWS & DevOps engineer	Nadimpalli Pradeep Kumar, Kanchana GM.
Approver Name and Designation	Soham Sen
Owning Department	AWS and DevOps

Contents:

1.Jenkins Continuous integration for ng-package-typing and package-typing-api:

Jenkins:

Jenkins is an open source, Java-based automation server that offers an easy way to set up a continuous integration and continuous delivery (CI/CD) pipeline.

Continuous Integration (CI) is the process of automating the build and testing of code every time a team member commits changes to version control. Continuous Delivery (CD) is the process to build, test, configure and deploy from a build to a production environment.

Git:

Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git Checkout:

The git checkout is navigator command that helps to switch branches. This option prepares you to work in a particular working branch. It Updates files in the working tree to match the version in the index or the specified tree. If no paths are given, git checkout will also update HEAD to set the specified branch as the current branch.

Npm:(Node Package Manager):

npm is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the world's largest software registry. Open-source developers from every continent use npm to share and borrow packages, and many organizations use npm to manage private development.

pm2:(process Manager 2):

Pm2 is an Open Source, advanced process manager for NodeJS applications that allows you quickly start, control, or stop your node processes. Some key features of PM2 are automatic application load balancing, declarative application configuration, deployment system and monitoring.

Pipeline:

Pipeline defines a "block" containing all content and instructions for executing the entire Pipeline.

Agent instructs Jenkins to allocate an executor (on a node) and workspace for the entire Pipeline.

stage is a syntax block that describes a stage of this Pipeline.

Steps is Declarative Pipeline-specific syntax that describes the steps to be run in this stage.

Sh is that executes the given shell command.

Node/slave:

When creating a new slave node, Jenkins allows us to tag a slave node with a label. Labels represent a way of naming one or more slaves. We leverage this labeling system to tie the execution of a job directly to one or more slave nodes.

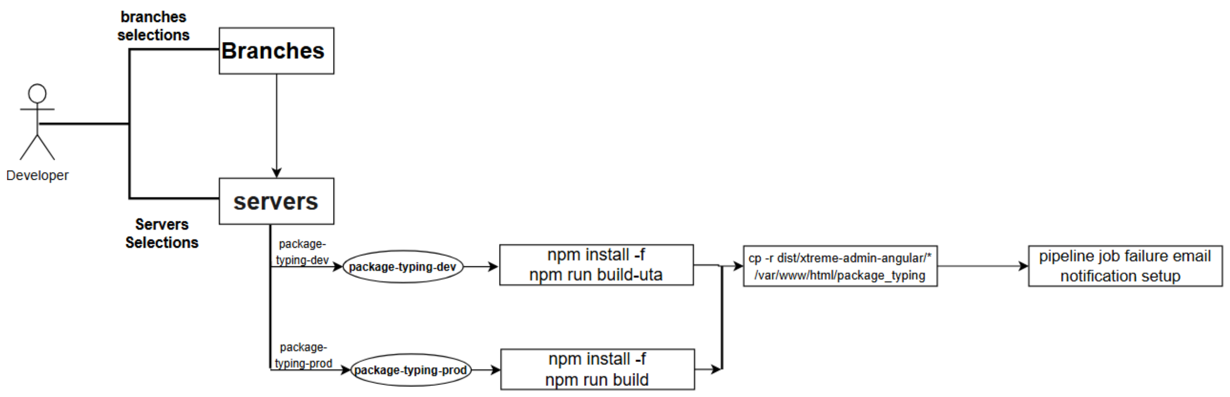
GitHub webhook / Build when a change is pushed to GitLab:

If Developers do modifications to code or merge or pushed code to specified branch the build will start immediately, and new code will reflect in mentioned server. (Automation Trigger)

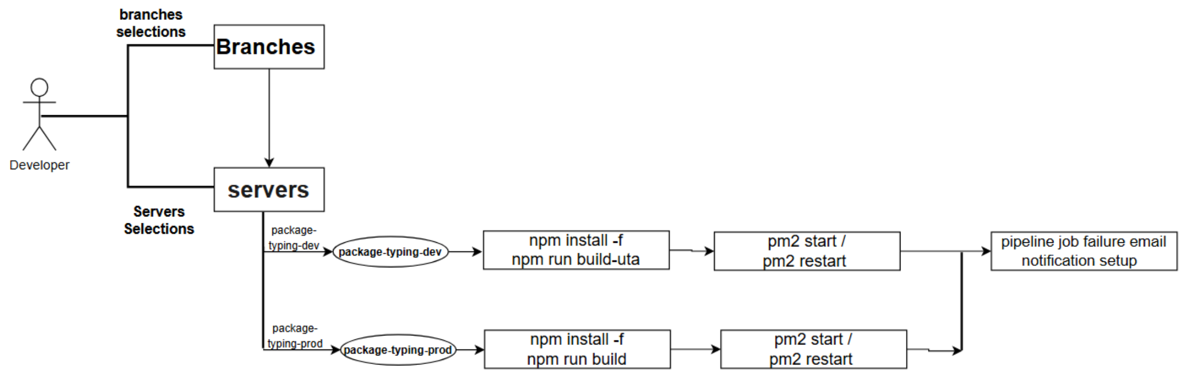
Email Notifications:

This email notifications setup will help Developers and DevOps engineers to get notifications for job build fail, success and unchanged, we will mention email notifications in post step in pipeline code.

1.1 Generic flow diagram of Jenkins pipeline for ng-package-typing:



1.2 Generic flow diagram of Jenkins pipeline for package-typing-api:



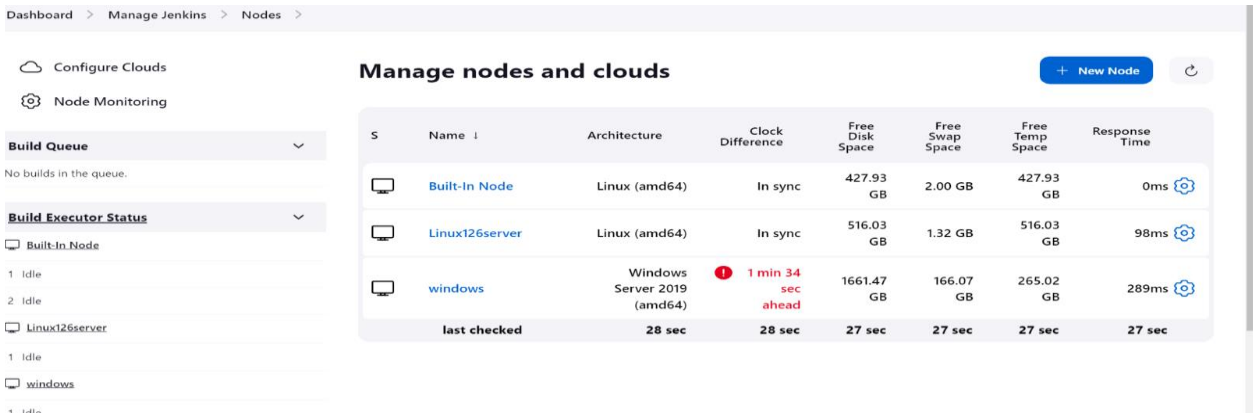
- By Using Jenkins, we create Pipeline job by Mentioning the Git Repository Clone URL https://proj.git of Project in the pipeline stage for Git Checkout.
- First time it clones the Repo and then from next time it pulls the latest code when job builds.
- In Jenkins configure we make a GitLab connection, and We set up email and extended email notifications by providing email suffix, port number (587 for Outlook) and email credentials.
- For Continuous automation we select build trigger tool in job configure based on our requirement
- By mentioning stages, we set up pipeline for above flow.

Plugins: Before working with CI pipeline job, we must install git, GitLab, email notifications plugins, git parameters

2.Project Requirement: ng package typing Repo Code Clone and for this project we are using build with parameters, using this we will get list of branches there in GitLab repo, we will select specified branch when a developer commits code to git branch.

Step1: Slave Connection

- We make a slave connection with prod server where we set Continuous Integration and if connection established successfully, we get this output.


Dashboard > Manage Jenkins > Nodes >

Configure Clouds
Node Monitoring

Build Queue
No builds in the queue.

Build Executor Status
Built-In Node
1 Idle
2 Idle
Linux126server
1 Idle
windows
1 Idle

Manage nodes and clouds

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	427.93 GB	2.00 GB	427.93 GB	0ms
	Linux126server	Linux (amd64)	In sync	516.03 GB	1.32 GB	516.03 GB	98ms
	windows	Windows Server 2019 (amd64)	1 min 34 sec ahead	1661.47 GB	166.07 GB	265.02 GB	289ms
	last checked	28 sec	28 sec	27 sec	27 sec	27 sec	27 sec

Windows and Linux(ubuntu) servers Added and connected to Jenkins.

Step2: Email Notification setup

Email Notifications:

This email notifications setup will help Developers and Devops engineers to get notifications for job build fail, success and unchanged, we will mention email notifications in post step in pipeline code.

If we set it for job failure, we will get email immediately to mentioned email addresses from Jenkins side by providing Console output history URL with job name and build number.
So that we can resolve the issue immediately.

The developers who are working on this project will get Jenkins email build fail notifications, we mention their mail id in To and CC in post pipeline stage.

Step3: This project is parameterized

For this project we selected This project is parameterized, under this we selected list git branches and choice parameter. Using this we will get build with parameters option, it will list the GitLab branches and Slave. When the new branches are creating automatically the newly created branches also update.

For the build job we got build with parameters option there we will select which branch developers want to build, and which server want to build

2.1 Flow Diagram of selecting branches and slave using Build with Parameters.

3. Pipeline code.

3.1 Pipeline code for ng-package-typing:

```
pipeline {

 agent none

tools {nodejs "npm"}

stages {

     stage('checkout') {

      agent {

        label "${params.SLAVE}"

      }

steps {

          echo "User selected branch is ${params.FROMBRANCH.split('/').last()}"

          git credentialsId: 'PAT', url: 'https://git.stldev.io:1305/angular/ng-package-
typing.git', branch: "${params.FROMBRANCH.split('/').last()}"


      sh " npm install"


    }

  }

stage('build'){
```

```
agent {
            label "${params.SLAVE}"
         }
       steps{
  script {
               if (params.SLAVE == 'package-typing-dev') {
                   sh 'npm run build-uat'
              } else if (params.SLAVE == 'package-typing-prod') {
                   sh 'npm run build'
}
            }
           }
      }
    stage('Copy dist'){
          agent {
            label "${params.SLAVE}"
         }
   steps{
          sh " cp -r dist/xtreme-admin-angular/* /var/www/html/package_typing "
}
}
}
}
```

Tools:

We will add the tools configuration under the system configuration in Jenkins. Tools help to integrate developer tools into the Jenkins environment.

First Stage: (Git checkout)

For this project we are using build with parameters, using this we will get list of branches and slaves, for that purpose we mentioned params.In this stage, it will clone the selected branch of https://git.stldev.io:1305/angular/ng-package-typing.git.To selected slave Jenkins working directory \\Jenkins\\workspace\\ ng-package-typing \\. To clone GitLab repository we will add GitLab credentials in jenkins.

Second stage:(npm install)

In this stage first it will check which server this stage needs to work. Then it will install the npm package.

Third stage:(build)

In this stage first it will check which server this stage needs to work. In this stage we are using if and else if statement. It will validate the server then it will run the npm run build command.

Fourt stage:(copying**)**

In this stage first it will check which server this stage needs to work. then it will copy the dist folder to html folder.

3.2 Pipeline code for package-typing-api:

```
pipeline {
        agent none
    tools {nodejs "npm"}
    stages {
       stage('checkout') {
         agent {
           label "${params.SLAVE}"
         }
         steps {
            echo "User selected branch is ${params.FROMBRANCH.split('/').last()}"
             git credentialsId: 'PAT', url: 'https://git.stldev.io:1305/angular/nest-package-
typing.git', branch: "${params.FROMBRANCH.split('/').last()}"
                        }
      }
      stage('copy workspace code to main dir path') {
          agent {
           label "${params.SLAVE}"
         }
         steps{


         sh " cp -rf /home/jenkins/workspace/package_typing_api /var/opt/ "
         }
      }
           stage('npm install ') {
```

```
          agent {
            label "${params.SLAVE}"
         }
          steps{
                dir('/var/opt/package_typing_api') {
                    sh " pwd "
                   sh " npm install "
                  sh " npm run build "
          }
       }
       }
        stage('pm2 start') {
          agent {
            label "${params.SLAVE}"
         }
          steps{
                dir('/var/opt/package_typing_api') {
                   script {
                 if (params.SLAVE == 'package-typing-dev') {
                     sh " /home/jenkins/.nvm/versions/node/v20.11.0/bin/pm2 start dev.config.js ||
true &&  /home/jenkins/.nvm/versions/node/v20.11.0/bin/pm2 restart all "
                 } else if (params.SLAVE == 'package-typing-prod') {
                     sh " pm2 start prod.config.js || true && pm2 restart all "
                 }
             }
          }
```

```
      }
      }
  }
```

Tools:

We will add the tools configuration under the system configuration in Jenkins. Tools help to integrate developer tools into the Jenkins environment.

First Stage: (Git checkout)

For this project we are using build with parameters, using this we will get list of branches and slaves, for that purpose we mentioned params.

In this stage, it will clone the selected branch of https://git.stldev.io:1305/angular/nest-package-typing.git. To selected slave Jenkins working directory \\Jenkins\\workspace\\ package-typing-api. To clone GitLab repository we will add GitLab credentials in jenkins.

Second stage:(copying workspace to main directory)

In this stage first it will check which server this stage needs to work. Then it will copy the file form jenkins working directory of /home/jenkins/workspace/ package-typing-api to var/opt/

Third stage: (npm install and build)

In this stage first it will check which server this stage needs to work. It will move to /var/opt/ package_typing_api directory, then it will install the npm and npm build

Fourth stage: (pm2)

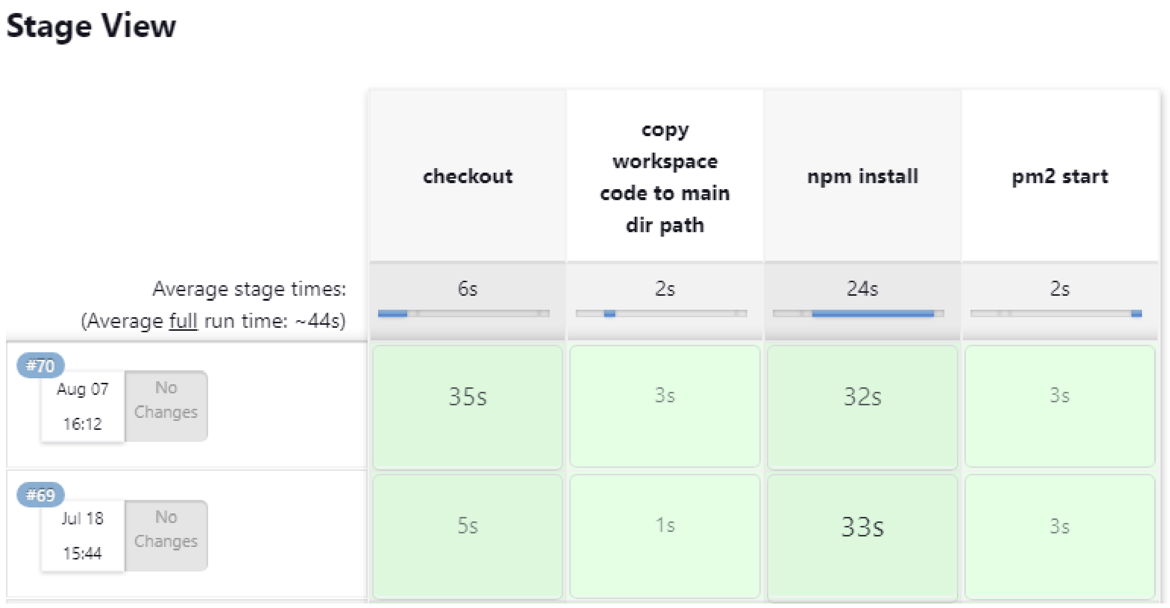
In this stage first it will check which server this stage needs to work. And it will move to /var/opt/ package_typing_api directory. In this stage we are using if and else if statement, it will validate the server, as for the APP-NAME it will start or restart the pm2

Now save the job

Build the Job by clicking on build with parameters and check console output.

Stage view of ng-package-typing:

Stage view of ng-package-typing-api:



Stage View

	checkout	copy workspace code to main dir path	npm install	pm2 start
Average stage times: (Average full run time: ~44s)	6s	2s	24s	2s
#70 Aug 07 16:12 No Changes	35s	3s	32s	3s
#69 Jul 18 15:44 No Changes	5s	1s	33s	3s

4. Flow of Jenkins CI to prod/dev server using Jenkins:

Jenkins Linux slave flow:

We will mention copy files from WS repo to specific repo, based on the requirement.

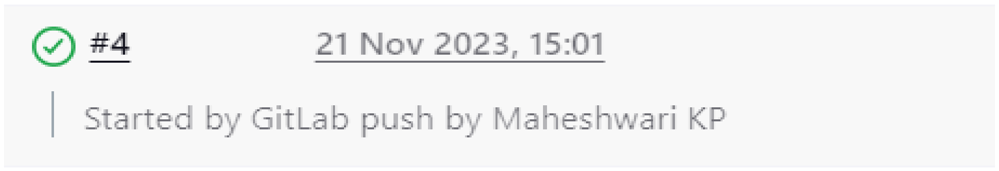
5.Jenkins Build Trigger for automating Job:

We are using Git Webhook, If Developers do modifications to code or merge or pushed code to specified branch the build will start immediately, and new code will reflect in mentioned server (Automation).

Closed Merge Request Events

We Copy Secret token of webhook and we'll add URL of job and secret token in Git Project Repo.

Now the Job builds automatically with the help of Build Trigger and without human Support it Works as Automated job.


#4 21 Nov 2023, 15:01

Started by GitLab push by Maheshwari KP

Note: As of now we are using build job manually.