

### **Setup Iptables -->**

```
# apt install iptables iptables-persistent
# iptables -L
# iptables -S
# vim /etc/iptables/rules.v4 --> Add rule to allow ssh from only ecosmob network
# iptables-restore < /etc/iptables/rules.v4 --> apply and start the service with given rule
# iptables-save > /etc/iptables/rules.v4 --> apply rule permanently
# systemctl restart docker
# iptables -L
```

### **Docker Installation →**

```
# apt install apt-transport-https ca-certificates curl gnupg2 software-properties-common
# curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb_release
-cs) stable"
# apt update
# apt install docker-ce
# apt install docker-ce-cli
# systemctl status docker
```

### **Docker-compose Installation →**

```
# curl -L
"https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname
-s)-$(uname -m)" -o /usr/local/bin/docker-compose
# chmod +x /usr/local/bin/docker-compose
# ls -l /usr/local/bin/docker-compose
# ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
# docker-compose --version
```

### **Mariadb 10 →**

```
# apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xF1656F24C74CD1D8
# add-apt-repository 'deb [arch=amd64]
http://mariadb.mirror.liquidtelecom.com/repo/10.4/debian buster main'
# apt update
# apt install mariadb-server mariadb-client
# systemctl status mariadb
#mysql_secure_installation -----> PENDING
```

### MongoDb → Simple basic Installation only

```
# wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -  
# echo "deb http://repo.mongodb.org/apt/debian buster/mongodb-org/4.4 main" | sudo tee  
/etc/apt/sources.list.d/mongodb-org-4.4.list  
# apt-get update  
# apt-get install mongodb-org  
# apt info mongodb-org  
# systemctl enable --now mongod  
# systemctl status mongod
```

**Reference** → <https://linuxize.com/post/how-to-install-mongodb-on-debian-10/>

### Redis container →

In the PCS Bitbucket repository we have a folder called “docker” in which we have another folder called redis containing Dockerfile to create a new docker image of redis.

ecoproddlr01-pcs / docker / redis		
Name	Size	Last commit
..		
Dockerfile	91 B	2 days ago
docker-compose.yml	206 B	2 days ago

Whenever we do any commit in this branch, it will build a new docker image of redis from this Dockerfile and then that image will be pushed on our development server automatically.

### On the development server we have our configuration files at path →

/home/ecodialer/pcs/redis

redis.conf → Default redis configuration file

We can edit configuration files as per our need restart our docker container manually on the server with latest image name (Defined In docker-compose file)

## Web Server container →

We have 2 folders api and web in which we have our source code of Lumen and Yii2  
We can clone the web repository and then edit the source code as per our need and then commit the changes.

In Web-server Bitbucket repository we have a folder called “docker” in which we have a web folder. In the web folder we have Dockerfile to create a new docker image of the web.



The image shows a screenshot of a Bitbucket repository's file list. At the top, there is a folder icon followed by a forward slash '/'. Below this is a table with three columns: 'Name', 'Size', and 'Last commit'. The table lists five items: three folders ('api', 'docker', 'web') and two files ('.gitignore' and 'Jenkinsfile'). Each folder entry has a folder icon, and each file entry has a document icon. The 'Last commit' column shows the time since the last commit for each item.

Name	Size	Last commit
api		2 days ago
docker		2 days ago
web		2 days ago
.gitignore	702 B	2 days ago
Jenkinsfile	4.35 KB	18 minutes ago

Whenever we do any commit, it will build a new docker image and then that image will be pushed on our development server automatically.

Now we will have new image on our development server with the name -  
dockerregistry.ecosmob.net:5000/ecodialer-web:{**BRANCH**}-{**BUILD\_ID**}

**On the development server we have our configuration files / folder at path →**  
/home/ecodialer/web

**config** → Folder containing all the configuration files related to Yii2

**env** → will be used as .env file for Lumen

**nginx.conf** → Default nginx configuration file

**default** → Nginx configuration file (Lumen and Yii2)

**supervisord.conf** → To run multiple processes inside container

We can edit these files as per our need and can simply restart our docker container manually on the server with latest image name (Defined In docker-compose file)

## RTPEngine Server container →

We have used the source code of rtpengine in our repository root folder named “rtpengine” → We can clone the PCS repository and then edit the source code of RTPEngine as needed and then commit the changes.

In the PCS Bitbucket repository we have a folder called “docker” in which we have another folder called rtpengine containing Dockerfile to create a new docker image of RTPEngine.



Name	Size	Last commit
docker		2 days ago
rtpengine		5 days ago
.gitignore	624 B	2020-09-29
Jenkinsfile	5.21 KB	27 minutes ago

Whenever we do any commit, it will build a new docker image and then that image will be pushed on our development server automatically.

Now we will have new image on our development server with the name -  
dockerregistry.ecosmob.net:5000/ecodialer-rtpengine:{**BRANCH**}-{**BUILD\_ID**}

**On the development server we have our configuration files at path →**  
/home/ecodialer/pcs/rtpengine

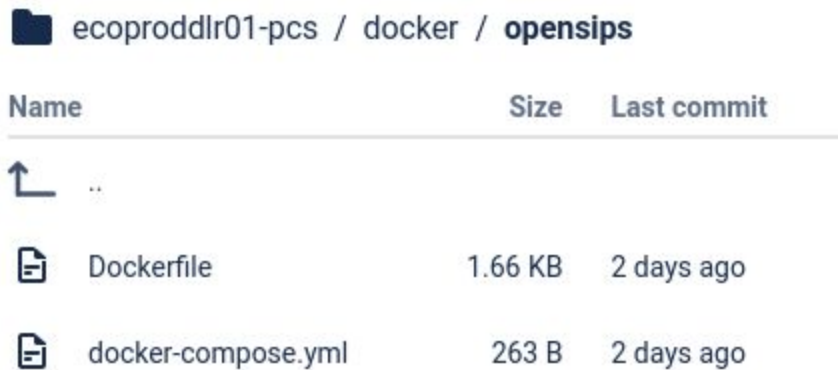
In the entrypoint.sh script we have defined “rtpengine --config-file /etc/rtpengine.conf” to start the rtpengine container and to pick “/etc/rtpengine.conf” configuration file.

We can edit the entrypoint.sh and rtpengine.conf as per our need

Now we can simply restart our docker container manually on the server with latest image name (Defined In docker-compose file)

## OpenSips Container →

In the PCS Bitbucket repository we have a folder called “docker” in which we have another folder called opensips containing Dockerfile to create a new docker image of opensips.



ecoproddlr01-pcs / docker / opensips		
Name	Size	Last commit
↑ ..		
📄 Dockerfile	1.66 KB	2 days ago
📄 docker-compose.yml	263 B	2 days ago

Whenever we do any commit, it will build a new docker image and then that image will be pushed on our development server automatically.

**On the development server we have our configuration files at path →**  
/home/ecodialer/pcs/opensips

**run.sh** → Provide command to run opensips “/usr/sbin/opensips -FE”

**opensips-default** → some default of opensips














**opensips-init** → main startup script for opensips

**opensips.cfg** → main configuration file for opensips

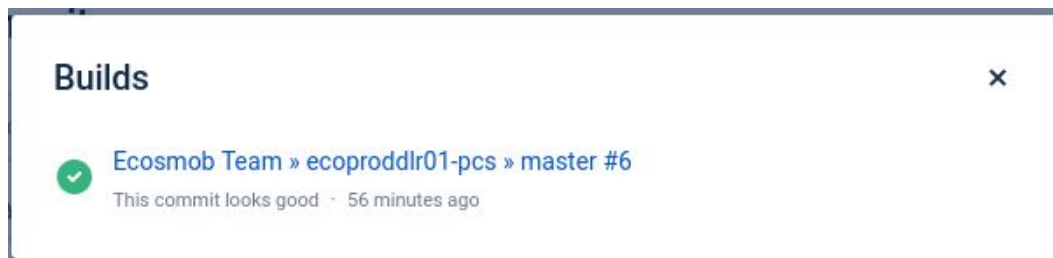
We can edit configuration files as per our need restart our docker container manually on the server with latest image name (Defined In docker-compose file)

To check the whole build status →

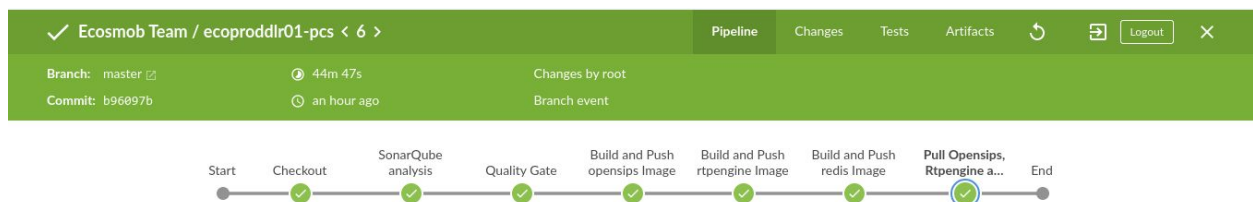
We can go to the bitbucket commit section, from there we can see the Builds, It will show us some status like green mark or red mark

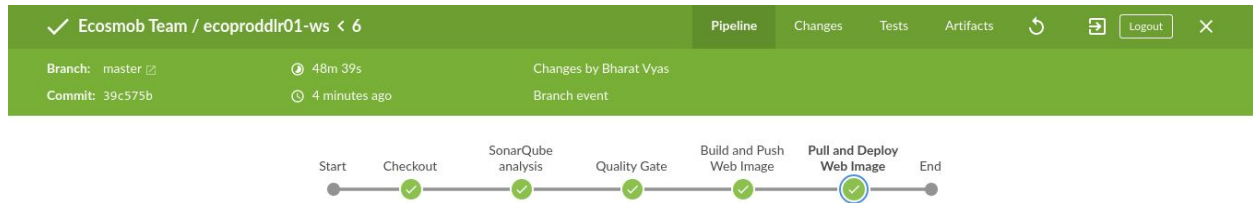
Commits					Clone
Search commits		All branches			
Author	Commit	Message	Date	Builds	
 Bharat Vyas	<a href="#">7216ce1</a>	Jenkinsfile edited online with Bitbucket	18 seconds ago		
 Bharat Vyas	<a href="#">b96097b</a>	edit Jenkinsfile and add redis folder	2 hours ago		
 Bharat Vyas	<a href="#">2c80cc3</a>	Jenkinsfile edited online with Bitbucket	6 hours ago		
 Bharat Vyas	<a href="#">5eb2423</a>	Exclude rtengine source	6 hours ago		
 Bharat Vyas	<a href="#">299f959</a>	Edit Jenkinsfile and add rtengine source	6 hours ago		
 Bharat Vyas	<a href="#">700798e</a>	Dockerfile edited online with Bitbucket	17 hours ago		
 Bharat Vyas	<a href="#">4fd34b8</a>	added Jenkinsfile and docker firectory	18 hours ago		

We can click on that green mark → It will show us another Build status, we can simply click on the mentioned link and it will open Jenkins blue ocean.



From there we can see the status of our build, and we can even check in which state it is failing





We can click on any of the stages and check the details of that as well.

## Email Notification →

We are using Jenkins to trigger email notifications for every successful build.  
We can change the recipient list as per our need in Jenkinsfile

In our Jenkinsfile we have a “Email Notification” stage something like →

```
stage("Email Notification")
{
    // mail bcc: '', body: 'This is test Email Notification !! ${env.JOB_BASE_NAME} or ${env.JOB_BASE_NAME} env.JOB_BASE_NAME ', cc: '', from: '', replyTo:
    emailx body: "${currentBuild.currentResult}: Job ${env.JOB_NAME} build ${env.BUILD_NUMBER}\n Git committer email: ${GIT_COMMIT_EMAIL} and ${GIT
    recipientProviders: [[class: 'DevelopersRecipientProvider'], [class: 'RequesterRecipientProvider']], subject: 'Jenkins Build Notification ', to
```

In this stage we have an option “to:” In which we can define the emails (comma separated) on which we want to send Email Notification

**Example →**

**to: 'user1@ecosmob.com,user2@gmail.com'**

And we have to change the IP address of remote server in Email section

We have provided “**Server IP**” in which we can provide our remote server IP and it will be shown in our email notification