# Versioning

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Modifications** |
| 1.0 | 7/3/2017 | Ahmed Misbah | Document created |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Installation Environment 3](#_Toc476589479)

[Minimum Requirements 3](#_Toc476589480)

[Recommended Requirements 3](#_Toc476589481)

[Installation Pre-requisites 4](#_Toc476589482)

[Installation/Setup of PostgreSQL RDBMS: 5](#_Toc476589483)

[Installation of PostgreSQL on Linux 5](#_Toc476589484)

[Installation of pgAdmin on a Windows machine 6](#_Toc476589485)

[Installation of Kafka Messaging Queue 8](#_Toc476589486)

[Installation of Couchbase: 9](#_Toc476589487)

[Installation of Microservices 10](#_Toc476589488)

[Install Oracle Java 10](#_Toc476589489)

[Installing JARS 11](#_Toc476589490)

[Install Web Application 13](#_Toc476589491)

[Install Apache2 httpd server 15](#_Toc476589492)

# Installation Environment

## Minimum Requirements

* 1 \* 32GB RAM (one chip) machine for hosting our micro-services for Development and testing environments
* 1 \* 16GB RAM (one chip) machine for hosting our DBs and message queues
* 1 \* 32GB RAM (one chip) machine for hosting Apache Cassandra (Big Data store)

## Recommended Requirements

* 1 \* 64GB RAM machine for Micro-services
* 1 \* 8 GB RAM machine API Gateway (Zuul) and Register Server (Eureka)
* 1\* 32 GB machine for Couchbase and Kafka queues
* 1 \* 4GB machine PostgreSQL RDBMS
* 1 \* 4GB machine for Static Content Server
* 1 \* 16GB machine for Apache Cassandra
* 1 \* 4GB machine for Apache httpd

# Installation Pre-requisites

Every machine should have a version of Ubuntu 16.04.1 LTS installed.

# Installation/Setup of PostgreSQL RDBMS:

## Installation of PostgreSQL on Linux

1. Download PostgreSQL using the following commands:

sudo apt-get update

sudo apt-get install postgresql postgresql-contrib

1. In order to use PostgreSQL, we'll need to first login to an account:

sudo -i -u postgres

1. You can get a PostgreSQL prompt immediately by typing:

psql

1. Change PostgreSQL user password using the following commands:

\password

Enter new password: root  
Enter it again: root

1. Then execute \q to exit
2. Access postgresql.conf file:

sudo nano /etc/postgresql/9.5/main/postgresql.conf

1. Update listen\_addresses attribute to be with no starting (#)

listen\_addresses = '\*'

1. Save file using ctrl+x then y and Enter.
2. Access pg\_hba.conf file:

sudo nano /etc/postgresql/9.5/main/pg\_hba.conf

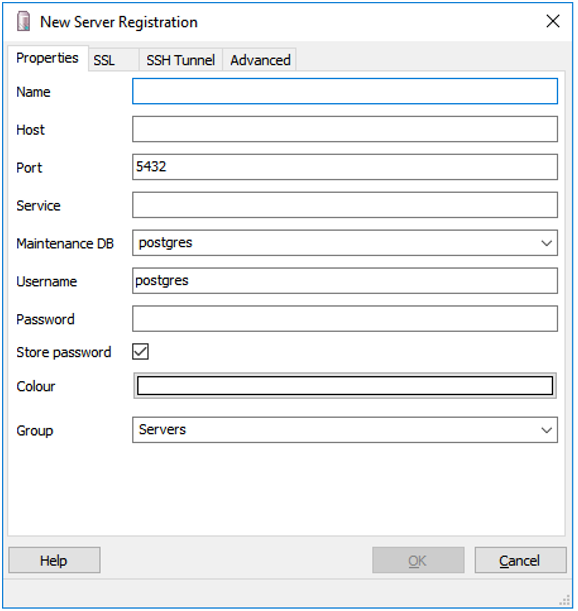
1. Change # IPv4 local connections to be

host all all 0.0.0.0/0 md5

1. Save file and exit
2. Execute: service.postgresql restart

## Installation of pgAdmin on a Windows machine

1. Using pgAdim client after download and install it from host <https://ftp.postgresql.org/pub/pgadmin3/pgadmin4/v1.2/windows/pgadmin4-1.2-x86.exe>
2. Double click to install.
3. When installation is done, open pgAdmin then open new server connection
4. Fill in server connection parameters



1. Right click on databases then enter “lastmile” on name field
2. Do it again with name “lastmile\_workflow”

# Installation of Kafka Messaging Queue

1. Download kafka\_2.11-0.10.1.0:

Wget <http://www-us.apache.org/dist/kafka/0.10.1.0/kafka_2.11-0.10.1.0.tgz>

1. Extract files using:

tar -xzf kafka\_2.11-0.10.1.0.tgz

1. Start service by executing:

bin/zookeeper-server-start.sh config/zookeeper.properties

bin/kafka-server-start.sh config/server.properties

# Installation of Couchbase:

1. Execute following commands to download and install Couchbase Version 4.5.0 :

wget <https://packages.couchbase.com/releases/4.5.0/couchbase-server-community_4.5.0-ubuntu14.04_amd64.deb>

sudo dpkg –I couchbase-server-community\_4.5.0-ubuntu14.04\_amd64.deb

1. After installation open browser then enter URL : <server url>:8091
2. Follow instructions to configure Couchbase:

* Specify 5000MB as couchbase memory.
* Configure to use value ejection.
* Uncheck default bucket installation
* Create new bucket with size 2048 and name “vehicle\_tracking”
* Create new bucket with size 2048 and name “search\_topic”
* Create new bucket with size 1024 and name “customer\_app\_firebase\_token”
* Click on query tab then execute commands:
* CREATE PRIMARY INDEX ON vehicle\_tracking
* CREATE PRIMARY INDEX ON search\_topic
* CREATE PRIMARY INDEX ON customer\_app\_firebase\_token

# Installation of Microservices

## Install Oracle Java

1. Execute following commands for download:

sudo apt-get install python-software-properties

sudo add-apt-repository ppa:webupd8team/java

sudo apt-get update

1. Execute following commands for installation:

sudo apt-get install oracle-java8-installer

1. Approve installation by clicking OK

sudo apt-get install oracle-java8-set-default

1. Configure JAVA\_HOME by Command

sudo nano /etc/environment

1. In this file, add the following line:

JAVA\_HOME=”/usr/lib/jvm/java-8-oracle”

1. Save file with overwriting it.
2. Test installation :
3. Print java home directory:

echo $JAVA\_HOME

1. Print java version used which is 1.8

java –v

## Installing JARS

1. Copy Microservices into your desired directory.
2. Define Service Discovery (Eureka) Machine IP, and Edge Server IP (ZUUL).
3. Each service in services directory has its own configuration file in SevicesConfig directory as <Service Name>-config.yml .
4. Modify configuration files as follows :

**Eureka:**

eureka:

client:

serviceUrl:

defaultZone: http://<Service Discovery IP>:8761/eureka/

instance:

instance-id: ${spring.cloud.client.hostname}:${spring.application.name}:${spring.application.instance\_id:${random.value}}

**Database:**

spring:

datasource:

url: jdbc:postgresql://<IP>:5432/lastmile

**Couchbase:**

couchbase:

bootstrap-hosts: <IP>

**Authorization Server:**

tokenServiceEndpointUrl:http://<IP>:60700/u/oauth/check\_token

**Kafka:**

kafka:

bootstrap:

servers: <IP>:9092

1. Update file lastMile\_services.sh set parameter service\_path with microservices directory, and config\_file\_path with service.config directory.
2. Execute command:

chmod +x ./lastMile\_services.sh

1. Execute:

./lastMile\_services.sh

1. Select option 1 “Standalone Services”
2. Select service number to install or 0 to install all .
3. Execute steps from 5 to 7 if you want to install service by service.
4. Wait until all services are up.

# Install Web Application

1. Install nodejs, execute command :

sudo apt-get install python-software-properties

curl -sL https://deb.nodesource.com/setup\_7.x | sudo -E bash -

1. Then execute :

**sudo apt-get install nodejs**

1. Check installation:

node -v

1. The output should be:

v7.5.0

1. Execute :

apt-get forever –g

1. Access Web application directory

sudo nano ~/<Frontend Project>/commons/js/constants.js and

1. Add edit following configurations:

angular.module('utilitiesModule', []).constant("absoluteURL", "https://<Zuul Server IP>:8080");

angular.module('utilitiesModule').constant("downloadURL", "https://<Static Content IP>:8082");

angular.module('utilitiesModule').constant("SOCKET\_URL", "wss://<Socket Service IP>:9761");

1. Execute the following command:

forever start server.js

1. Open browser and enter URL:

<https://localhost:3000>

or replace localhost with machine’s IP address.

# Install Apache2 httpd server

1. Install Apache2 using the following commands:

sudo apt-get update

sudo apt-get install apache2

sudo a2enmod

deflate headers lbmethod\_byrequests proxy proxy\_ajp proxy\_balancer proxy\_connect proxy\_http proxy\_wstunnel rewrite ssl

1. Overwrite sites\_enable directory with our configured one
2. configure 000-default.conf with urls of proxied servers
3. Change SSL configuration with our certificates directory:

SSLCertificateFile "certificates/ssl.crt/ca.crt"

SSLCertificateKeyFile "certificates/ssl.key/ca.key"

1. Change balancer URl to use VehicleRegistration microservice machine IP URL BalancerMember ws://<IP>:10590
2. Configure RewriteRule configurations to map to OpenSocket service IP

RewriteRule /1 ws://<IP>:%1 [P]

ProxyPassReverse /1 ws://<IP>:%1

1. Copy certificate directory to apache2 directory
2. Add logs directory to /apache2 directory
3. Execute:

sudo service apache2 restart