**Logmonitor** – Centralized log monitoring system

Follow below steps for installation –

1. Install Postgres with PGAdmin 3/ PGAdmin 4

Create DB with name LogMonitor with schema thrdexams

Create thrdexams schema inside LogMonitor DB

1. Download the repository from github extract on your local drive and import it into STS 4.
2. Docker for Windows

<https://docs.docker.com/desktop/install/windows-install/>

download the installer from above link and follow the installation steps asked during installation. It will ask to reboot the machine click yes.

Note – If asked to update WSL click yes.

1. If docker is installed successfully run the below command to start RABBITMQ inside docker

docker run -d --name rabbitmq -p 5672:5672 -p 5673:5673 -p 15672:15672 rabbitmq:3.11.9-management

1. Open the Logmonitor project in STS4. Do maven update.

If it is giving log not found or any log related error install Lombok to your sts.

If still it is given error delete/rename .m2 folder

1. Update the application.properties in client,master and db services as mentioned below

Logmonitor-Client

Client support POSTGRES and APACHE tomcat

For APACHE use the below option

logmonitor.server-type=APACHE

For POSTGRES use the below option

logmonitor.server-type=POSTGRES

logmonitor.projectname=CIPET

logmonitor.logpath=D:\\installers\\PostgreSQL\\data\\pg\_log

logmonitor.linematcher=ERROR

logmonitor.interval=20000

server.port=9000

logmonitor.server-type=POSTGRES

Logmonitor-Master

Note – While using it in live keep spring.jpa.hibernate.ddl-auto=none

spring.datasource.url=jdbc:postgresql://localhost:5432/LogMonitor

spring.datasource.username=logmonitor

spring.datasource.password=logmonitor

spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect

spring.jpa.hibernate.ddl-auto=create-drop

spring.jpa.show-sql=true

server.port=9001

logmonitor.savetodb-interval=10000

logmonitor.savetodb-initialdelay=20000

LogMonitor-DBServices

spring.datasource.url=jdbc:postgresql://localhost:5432/LogMonitor

spring.datasource.username=logmonitor

spring.datasource.password=logmonitor

#spring.jpa.database-platform=org.hibernate.dialect.PostgreSQLDialect

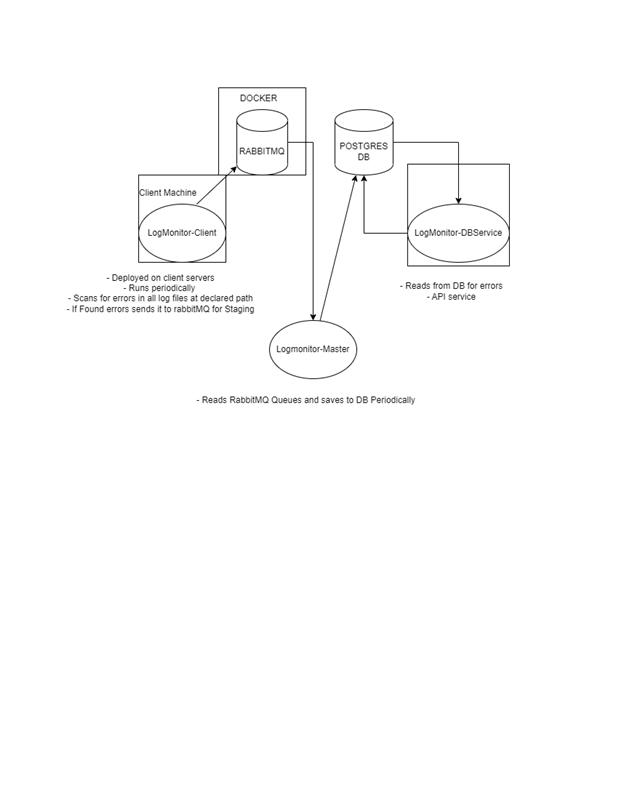
spring.jpa.hibernate.ddl-auto=none

spring.jpa.show-sql=true

server.port=9002

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

Overview -



Explanation –

Client scans the logpath periodically if found any value mentioned in linematcher then holds the errored line in array and send it to RabbitMQ for stagging. If the same error is found it doesn’t sends the request to RabbitMQ. In case of restart it will resend the same error request to RabbitMQ.

Note – While starting Logmonitor-Client do check the below line is getting printed on console.

o.s.a.r.c.CachingConnectionFactory [0;39m [2m:[0;39m Created new connection: rabbitConnectionFactory#11787625:0/SimpleConnection@49560775

It is making connection to RabbitMQ server.

Master reads the queues from RABBITMQ stores it in array and periodically saves the errored data in DB.

DBServices interact with DB and fetch the records based on endpoints.