# Steps to setup Hygieia Dashboard from Source Code.

* 1. Installation:

1. Download the source code from GitHub repository: <https://github.com/capitalone/Hygieia>

OR

Download the modified code which includes collector for Rally Feature collector, Splunk Collector, Selenium collector, changes to API and UI from Github repository: <https://github.com/saurav4342/Hygieia-Devops-Dashboard>

1. Download and Install latest version of MongoDB on your system

Download URL: <https://www.mongodb.com/download-center>

1. Download and Install latest version of Maven in your system and set the appropriate class Path in system variables.

Download URL: <https://maven.apache.org/download.cgi>

1. Check if a proxy network is configured on your system. If a proxy network is present, you have to declare the proxy server setting in Maven configuration file **settings.xml** .

Check this guide for further reference: <https://maven.apache.org/guides/mini/guide-proxies.html>

b. DB Configuration:

1. Create a folder in the Drive where Hygieia is to be installed to act as the data storage space for mongo server. Go to the bin directory of your mongodb installation and run the following command to start the mongodb. Do make sure that data directory should pre-exist at the target location   
   mongod --dbpath < path to the data directory>    
   for e.g  /usr/bin/mongodb-linux-x86\_64-2.6.3/bin/mongod --dbpath /dev/data/db
2. Run the following commands as shown below at mongodb command prompt  /usr/bin/mongodb-linux-x86\_64-2.6.3/bin/mongo

$ mongo

MongoDB shell version: 3.0.4

connecting to: test

> use dashboard

switched to db dashboard

> db.createUser(

{

user: "dashboarduser",

pwd: "dbpassword",

roles: [

{role: "readWrite", db: "dashboard"}

]

})

Output similar to below should be seen in your mongo shell

Successfully added user: {

"user" : "dashboarduser",

"roles" : [

{

"role" : "readWrite",

"db" : "dashboard"

}

]

}

We recommend that you download MongoDB clients (RoboMongo etc) to connect to your locally running database and make sure that database: dashboard is created, and you are able to connect to it.

c. API installation and configuration:

1. From the Command prompt, go to API directory. Run mvn install to package the API code into an executable JAR file. You’ll notice a new directory has been created named “Target” and a file api.jar created inside the target directory.
2. Create a file named application.properties inside the Target folder and define the configuration properties of the application in that file. Following are the configuration parameters that can be defined. Parameters numbered 1 till 5 cannot be empty. Rest can be ignored.
3. dbname=dashboard
4. dbusername=[MogoDb Database Username, defaults to empty]
5. dbpassword=[MongoDb Database Password, defaults to empty]
6. dbhost=[Host on which MongoDb is running, defaults to localhost]
7. dbport=[Port on which MongoDb is listening, defaults to 27017]
8. dbreplicaset=[false if you are not using MongoDB replicaset]
9. dbhostport=[host1:port1,host2:port2,host3:port3]
10. server.contextPath=[Web Context path if any]
11. server.port=[Web server port - default is 8080]
12. corsEnabled=false
13. corsWhitelist=http://domain1.com:port,http://domain2.com:port
14. version.number=@application.version.number@
15. auth.expirationTime=[JWT expiration time in milliseconds]
16. auth.secret=[Secret Key used to validate the JWT tokens]
17. # LDAP Server Url, including port of your LDAP server
18. auth.ldapServerUrl=[ldap://company.com:389]
19. # If using standard ldap
20. # LDAP User Dn Pattern, where the username is replaced with '{0}'
21. auth.ldapUserDnPattern=[uid={0},OU=Users,dc=your,dc=company,dc=com]
22. # If using ActiveDirectory
23. # This will be the domain part of your userPrincipalName
24. auth.adDomain=[company.com]
25. # This will be your root dn
26. auth.adRootDn=[dc=your,dc=company,dc=com]
27. # This will be your active directory url (required for AD)
28. auth.adUrl=[Need an example]
29. monitor.proxy.host=[hostname of proxy server]
30. monitor.proxy.type=[http|socks|direct]
31. monitor.proxy.port=[port enabled on proxy server]
32. monitor.proxy.username=[proxy username]
33. monitor.proxy.password=[proxy password]
34. Run the api.jar file created inside the target directory by typing the following in command prompt:

java -jar api.jar -Djasypt.encryptor.password=hygieiasecret

For more information on configuration see the spring boot documentation URL: <http://docs.spring.io/spring-boot/docs/current-SNAPSHOT/reference/htmlsingle/#boot-features-external-config-application-property-files>

1. Verify if your api is running by hitting the following URL in your browser : <http://localhost:8080/api/dashboard>

If you receive an empty array in the form of JSON response than you are all good for further installation. If you get an error in loading the URL please check the host/port on which you are running the api in application.properties.

d. Collector installation and Configuration

1. Analyze the collectors you need to configure for your application. Go inside each collector and build the collector using mvn install from command prompt.

Target directory gets created inside the collector directory and jar file is built.

Create an application.properties file for each collector having the configuration parameters required to run the application. Run the collector using the following command using command prompt:

Java –jar collector-name.jar

The collector should boot up and start running.

Application.properties file for each collector:

**Rally:**

# Database Name

dbname=dashboard

# Database HostName - default is localhost

dbhost=localhost

# Database Port - default is 27017

dbport=27017

# MongoDB replicaset

dbreplicaset=false

dbhostport=localhost:27017

# Database Username - default is blank

dbusername=your username

# Database Password - default is blank

dbpassword=your password

# Logging File location

logging.file=./logs/rally.log

# Chron schedule: S M D M Y [Day of the Week]

feature.cron=0 0/5 \* \* \* \*

# Rally CONNECTION DETAILS:

feature.rallyBaseUri=http://rally1.rallydev.com

feature.rallyApiKey=API key for your CA Rally account.

feature.projectObjectId=Project Object Id/Ref Id for one of the projects.

To find out your API key, go to this URL:

<https://help.rallydev.com/rally-application-manager>

**Jenkins:**

#Database Name - default is test

spring.data.mongodb.database=dashboard

#Database HostName - default is localhost

spring.data.mongodb.host=127.0.0.1

#Database Port - default is 27017

spring.data.mongodb.port=27017

#Database Username - default is blank

spring.data.mongodb.username=

#Database Password - default is blank

spring.data.mongodb.password=

#Collector schedule (required)

jenkins.cron=0 0/5 \* \* \* \*

#Jenkins server (required) - Can provide multiple

jenkins.servers[0]=server URL for jenkins

#If using username/token for api authentication (required for Cloudbees Jenkins Ops Center) see sample

#jenkins.servers[1]=

#Another option: If using same username/password Jenkins auth - set username/apiKey to use HTTP Basic Auth (blank=no auth)

jenkins.username=

jenkins.apiKey=

#Determines if build console log is collected - defaults to false

jenkins.saveLog=false

**SVN:**

dbname=dashboard

# Database HostName - default is localhost

dbhost=localhost

# Database Port - default is 27017

dbport=27017

# MongoDB replicaset

dbreplicaset=[false if you are not using MongoDB replicaset]

dbhostport=[host1:port1,host2:port2,host3:port3]

# Database Username - default is blank

dbusername=

# Database Password - default is blank

dbpassword=

# Logging File location

logging.file=./logs/subversion.log

# Collector schedule (required)

subversion.cron=0 0/1 \* \* \* \*

# Shared subversion username and password

subversion.username=svn username

subversion.password=svn password

subversion.url=svn host URL (for e.g. http://svn-mas.es.oneadp.com/svn/TLM\_ezlm)

# Maximum number of days to go back in time when fetching commits

subversion.commitThresholdDays=15

subversion.key=

**Splunk Collector:**

Before running Splunk Collector, first create a new saved search in your splunk application with the name “Hygieia” and with the following query :

index=ezlm\_main (loglevel=error OR loglevel=fatal) NOT PFC NOT "The specified associate does not have any active positions that are configured for mobile access" NOT eventtype=junklogs | eventstats count as TotalCount | stats last(version) as Version values(Msg) as ErrorMessage values(EZ\_CID) as uCIDs count as Errors values(SYSCODE) as SysCode max(TotalCount) as Total by pod | eval "%"=round(100\*Errors/Total,2) | fields - Total | sort - Errors

#Database Name - default is test

spring.data.mongodb.database=dashboard

#Database HostName - default is localhost

spring.data.mongodb.host=127.0.0.1

#Database Port - default is 27017

spring.data.mongodb.port=27017

#Database Username - default is blank

spring.data.mongodb.username=

#Database Password - default is blank

spring.data.mongodb.password=

#Collector schedule (required)

splunk.cron=0 0/1 \* \* \* \*

splunk.username=splunk login username

splunk.password=splunk login password

splunk.url=splunk host url

#Name of the application in splunk

splunk.app=name of your application as given in splunk(for e.g ezlm\_main)

splunk.savedSearch=Hygieia