Application Information Document

Document Id: Job Management Service

Job Management Service

Document Control Section

Version Number: 1.0

Authors

Role	Name
Application Developer	Dinesh Kori
Application Architect	Dinesh Kori

Reviewers

Role	Name

Revision History

Version #	Description of Changes	Issue Date
0.1	Initial Draft	29 Jan 2018

Table of Contents

1. OBJECTIVES

The objective of the document is an overview of the Job Management Service application. The document describes the function of the application, the structure of the applications, the application configuration and the technical environment, application Building, Application Running and smoke testing information.

2. TERMINOLOGY AND ACRONYMS

Acronyms and terminology specifically used in this document are described below.

No.	Terminology / Acronyms	Definition
1	AID	Application Information Document

3. APPLICATION OVERVIEW

Job Management service is simple J2EE application that takes job from REST API, schedule them for execution and report the status of Job.

Contacts

Name	Role/Application	Contact info
Dinesh Kumar Kori	Application Developer	Dinesh.kori86@gmail.com

4. ASSUMPTIONS

S.no.	Assumption	Remarks
1	System will always execute Job based on priority for system date.	
2	Returning status of Job is responsibility of Class implementing “Job” abstract class.	
3	Job Executor pool could be configured using application.properties	
4	Maximum job to be executed by Executor Services should be mentioned in application.properties	
5	Lower integer number is low Priority of job, higher number will give higher priority	
6	At a time, Job could have only one of the status from QUEUED, RUNNING, SUCCESS, FAILED	
7	Time to start execution on the day of execution is not considered so far in this release	Till now only Priority is considered.
8	Application need 9080 default port for starting.	Could be changed in application.properties
9	Job StartDate should be defined in dd/MM/yyyy	Time is not considered so far in this release

5. DEFINITION OF JOB & OTHER ASUMPUTIONS

JOB:

- This class implements Comparable to compare the Job Object based on Priority.
- This class implements Callable to return Future for the status of executing Job.
- Implements IJobs which define @Transaction Method to be implemented by Job Implementer
- Each Job should be declared "isconfigurable" if different set task needed to execute and should be defined in property file using helper class.
  Like CSVhelper1.properties for one of the CSV Job.

6. APPLICATION PLATFORM

Software Details

Technology	Product Name	Product Version	Vendor Name
Java	JDK	1.8.0_xx	Any
Application Server	Tomcat embedded in Spring Boot	9.0	Apache
Build Tool	Maven	2.2.x (or Above)	Apache
Codebase	GitHub	1.8	Git

API Details

SITES	URLs
Swagger URL	http://${HOSTNAME}/swagger-ui.html

Code URL

Code Repository	URL
GitHub	https://github.com/dineshkori/optile.git

7. HIGH LEVEL ARCHITECTURE

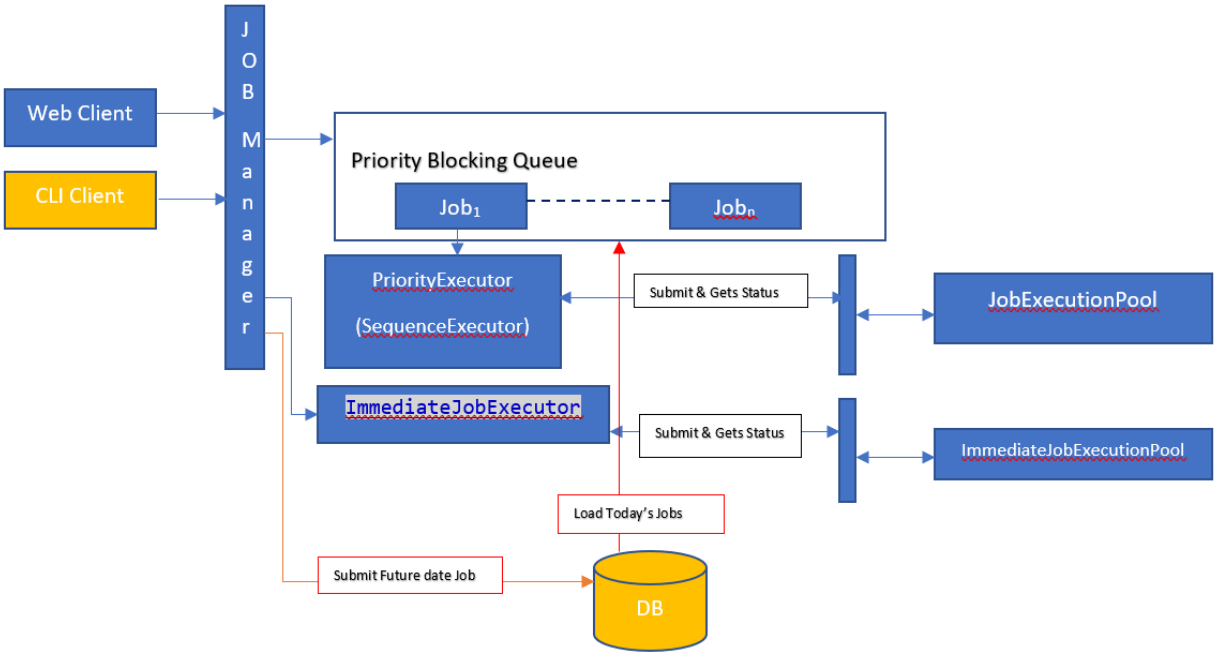


Figure 1: High Level Diagram

7.1 Client Application

Application provide Web REST API to submit the Jobs to the Job manager, future release could support CLI (Command Line Interface) also.

7.2 Job Manager

Job Manager is responsible for follow task in application

- Gets the Job from Client application
- Identify the job Type based on the Implementation of Job
- Identify if Job need immediate execution, to execute later in current system date or in future date then put it in appropriate Priority Queue or Immediate Execution or to be persisted so that could be taken in future.
- Poll the job from Priority Queue and assign it to execution of Job.
- Get the status of Job from Executor Services.

7.3 Database [Future Enhancement]

Any database could be used which will save all the Jobs, so that they can be polled on System current date for execution.
Jobs could be saved after execution also for reporting purpose.

7.4 Execution Poll

There are mainly two kind of Executor Pool which are based on java Executor Service that will execute the Job and will provide the status of Job as implements using java Future API.

7.4.1 Immediate Execution

Job need immediate Execution irrespective of priority.

7.4.2 Scheduler Execution

Polls a Job from Priority Blocking Queue and execute it.

8. APPLICATION STRUCTURE

8.1 Application Modules

Application is divided into below modules

8.1.1 baseJobs

This module is for all Job, this define Job Base class which is parent class for all the Jobs to be implemented. Below is some code snippet of this class.

```
public abstract class Job implements IJobs, Comparable<Job>, Callable<String> {

	/**
	 * Will Define the Status of Job i.e QUEUED, RUNNING, SUCCESS, FAILED
	 */
	private String status;

	/**
		 * Human readable Name for the Job


	........

	public int compareTo(Job other) {
		if (this.getPriority() > other.getPriority()) {
			return 1;
		} else if (this.getPriority() < other.getPriority()) {
			return -1;
		} else {
			return 0;
		}
	}

	}
```

8.1.2 JobImplementor

This module is dependent on baseJob module. This is the module which is responsible for actual implementation of any Job.
Below is the code snippet of sample CSV

```
public class CSVServiceImpl extends Job {

	public String call() {
		return this.runnner();
	}

	public String runnner() {
		this.setStatus(RUNNING);
```

```
            // TODO: Task related Operation
            } catch (JobExecutionException e) {
                  System.err.println("CSVJobs runnner Method Exception");
                  this.setStatus(FAILED);

            } finally {
                  // TODO: handle finally clause
                  if (this.getStatus().equalsIgnoreCase(FAILED) &&
this.isJobRollable()) {
                        this.rollbackjob();
                        return FAILED;
                  }
            }
            return SUCCESS;
      }

      public void rollbackjob() {
            System.err.println("CSVJobs rollbackjob Method");
            this.setStatus(FAILED);
      }

      /**
             * This is used to Set some configuration for this Job and should be
      using some properties file so that each Job could use specific configuration

       * @throws JobExecutionException
       */
      public void initialJobContext() throws JobExecutionException {
            if (this.isConfigureable()) {
                  // TODO Do Configuration before running this job
            }
      }

}
```

Helper Class

A helper class should be written like below for initializing context for job

```
public class CSVJobHelper implements IJobshelper {

      public void initialJobContext(String configFileName) throws
JobExecutionException {
            // TODO Read configFileName given in parameter and initialize this CSV
job
            // Context
            System.out.println("Yet to be implemented");
            System.out.println("No Job Context Set till now");
      }

}
```

Sample Job Config file

JobManagerServiceImpl.java | SampleCSVHelper1.properties

```
#Sample Property file which should be used for Configuring the CSV job Instance

#InfileLocation=
#OutFileLocation=
```

SampleCSVHelper1.properties

8.1.3 scheduler

Is the module which has all the implementation of exposing and scheduling task and executing them. This exposes the REST API as of now to submit the job.

Executables and Libraries

Executable	Purpose
Scheduler	"scheduler-0.0.1-SNAPSHOT.jar" is jar which could be executed for running Job Management services

9. BUILD PROCESS

9.1 Pre-requisite

You need to have JAVA_HOME & MAVEN_HOME set to build this application

E.g. for Windows
set JAVA_HOME=C:\Program Files\Java\jdk1.8.0_181
set MAVEN_HOME=C:\Projects\Ecom_Dev_HardRock_FE\apache-maven-2.2.1
set PATH=%PATH%;%JAVA_HOME%\bin;%MAVEN_HOME%\bin;
set PROJECT_HOME=${PROJECT_FOLDER_LOCATION}

Note: Location of JAVA_HOME & MAVEN_HOME can change based on your machine.

9.2 Building Module

Building process is maven projects

9.2.1 Build "baseJobs"

mvn -f %PROJECT_HOME%\baseJobs\pom.xml clean install package

This will build and install "baseJobs-0.0.1.jar" and install it into your local maven repository.

9.2.2 Build "jobImplementor"

mvn -f %PROJECT_HOME%\JobImplementor\pom.xml clean install package
This will build and install "JobImplementor-0.0.1.jar" and install it into your local maven repository.

9.2.3 Build "jobImplementor"

mvn -f %PROJECT_HOME%\ scheduler\pom.xml clean install package
This will build and install "scheduler-0.0.1-SNAPSHOT.jar" is a Spring Boot Executable file .

10. RUNNING APPLICATION

After successful build process, to run application use below command

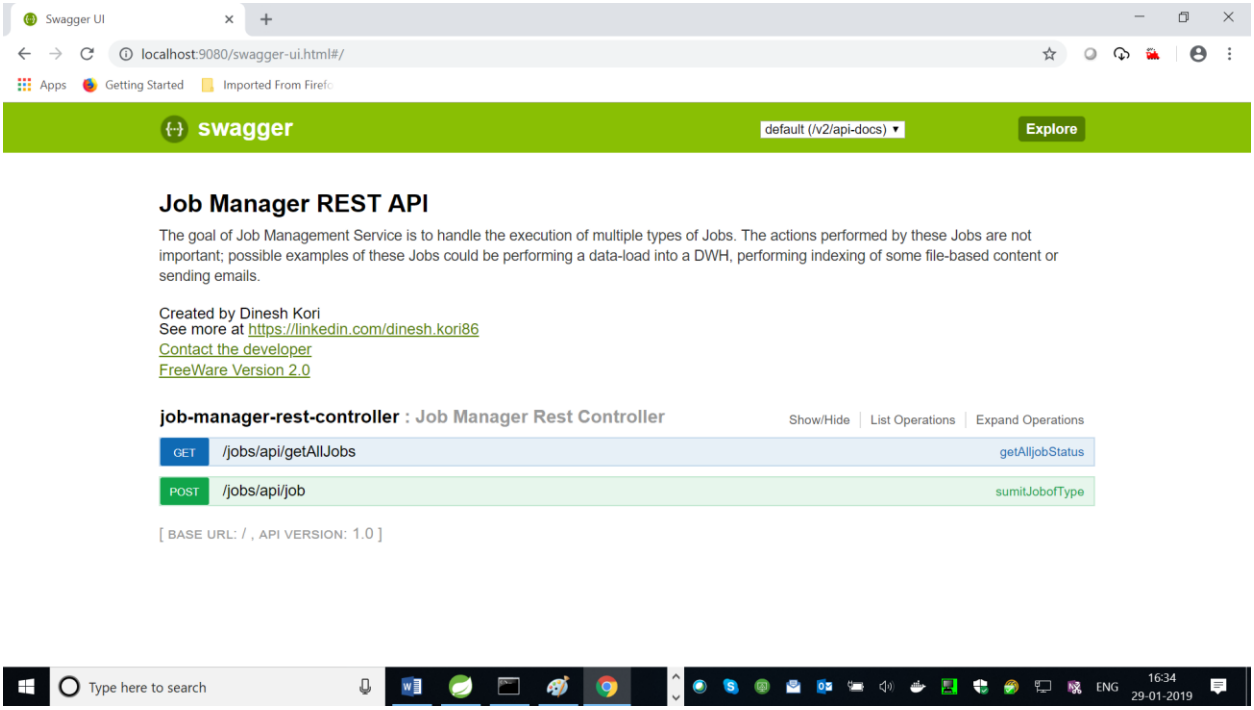
java -jar ${LOCATION}\ scheduler-0.0.1-SNAPSHOT.jar

Startup logs : you will be able to see the application running on port 9080

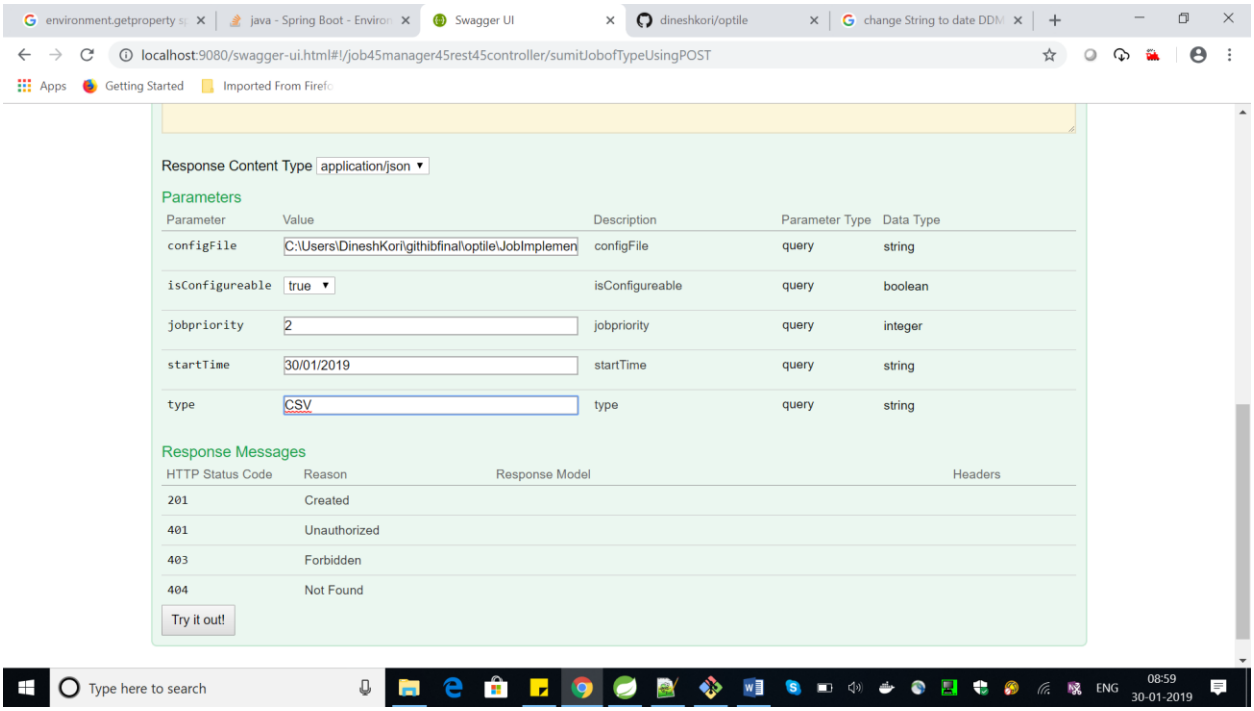
```
:\Users\DineshKori\githibfinal\optile\scheduler\target>java -jar scheduler-0.0.1-SNAPSHOT.jar

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::        (v2.1.1.RELEASE)

2019-01-29 16:29:02.239  INFO 5800 --- [           main] com.manager.job.Application              : Starting Application v0.0.1-SNAPSHOT on dinekori with PID 5800 (C:\Users
\DineshKori\githibfinal\optile\scheduler\target\scheduler-0.0.1-SNAPSHOT.jar started by DineshKori in C:\Users\DineshKori\githibfinal\optile\scheduler\target)
2019-01-29 16:29:02.248  INFO 5800 --- [           main] com.manager.job.Application              : No active profile set, falling back to default profiles: default
2019-01-29 16:29:06.864  INFO 5800 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 9080 (http)
2019-01-29 16:29:06.947  INFO 5800 --- [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2019-01-29 16:29:06.948  INFO 5800 --- [           main] org.apache.catalina.core.StandardEngine  : Starting Servlet Engine: Apache Tomcat/9.0.13
2019-01-29 16:29:06.973  INFO 5800 --- [           main] o.a.catalina.core.AprLifecycleListener   : The APR based Apache Tomcat Native library which allows optimal performa
nce in production environments was not found on the java.library.path: [C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\WINDOWS\Sun\Java\bin;C:\WINDOWS\system32
;C:\WINDOWS;C:\ProgramData\DockerDesktop\version-bin;C:\Program Files\Docker\Docker\Resources\bin;C:\oraclexe\app\oracle\product\11.2.0\server\bin;;C:\Program Files (x86)\C
ommon Files\Oracle\Java\javapath;C:\Program Files (x86)\RSA SecurID Token Common;C:\ProgramData\Oracle\Java\javapath;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem
;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C:\ProgramData\Webex\Webex\Applications;C:\Program Files\Git\cmd;C:\Program Files\helm\windows-amd
64;C:\Program Files\IBM\Cloud\bin;C:\Program Files\nodejs\;C:\Program Files\Microsoft VS Code\bin;C:\Program Files\TortoiseSVN\bin;C:\ProgramData\DockerDesktop\version-bin;
:\Program Files\Docker\Docker\Resources\bin;C:\oraclexe\app\oracle\product\11.2.0\server\bin;;C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Program Files (x8
6)\RSA SecurID Token Common;C:\ProgramData\Oracle\Java\javapath;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDO
WS\System32\OpenSSH\;C:\ProgramData\Webex\Webex\Applications;C:\Program Files\Git\cmd;C:\Program Files\helm\windows-amd64;C:\Program Files\IBM\Cloud\bin;C:\Program Files\no
dejs\;C:\Program Files\Microsoft VS Code\bin;C:\Program Files\TortoiseSVN\bin;C:\Program Files\Java\jdk1.7.0_80\bin;C:\ATT KT\SW\apache-maven-3.6.0-bin\apache-maven-3.6.0\b
in;C:\Program Files\Java\jdk1.7.0_80\bin;C:\ATT KT\SW\apache-maven-3.6.0-bin\apache-maven-3.6.0\bin;;C:\Users\DineshKori\AppData\Local\Programs\Fiddler;.]
2019-01-29 16:29:07.362  INFO 5800 --- [           main] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicationContext
2019-01-29 16:29:07.362  INFO 5800 --- [           main] o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization completed in 4977 ms
2019-01-29 16:29:08.474  INFO 5800 --- [           main] pertySourcedRequestMappingHandlerMapping : Mapped URL path [/v2/api-docs] onto method [public org.springframework.h
ttp.ResponseEntity<springfox.documentation.spring.web.json.Json> springfox.documentation.swagger2.web.Swagger2Controller.getDocumentation(java.lang.String,javax.servlet.htt
p.HttpServletRequest)]
2019-01-29 16:29:08.846  INFO 5800 --- [           main] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService 'applicationTaskExecutor'
2019-01-29 16:29:09.234  INFO 5800 --- [           main] o.s.s.c.ThreadPoolTaskScheduler          : Initializing ExecutorService 'taskScheduler'
2019-01-29 16:29:09.342  INFO 5800 --- [           main] d.s.w.p.DocumentationPluginsBootstrapper : Context refreshed
2019-01-29 16:29:09.428  INFO 5800 --- [           main] d.s.w.p.DocumentationPluginsBootstrapper : Found 1 custom documentation plugin(s)
2019-01-29 16:29:09.502  INFO 5800 --- [           main] s.d.s.w.s.ApiListingReferenceScanner     : Scanning for api listing references
2019-01-29 16:29:10.057  INFO 5800 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 9080 (http) with context path ''
2019-01-29 16:29:10.068  INFO 5800 --- [           main] com.manager.job.Application              : Started Application in 8.679 seconds (JVM running for 9.683)
```

11. TEST APPLICATION

To test application, you use Swagger UI to perform sanity testing

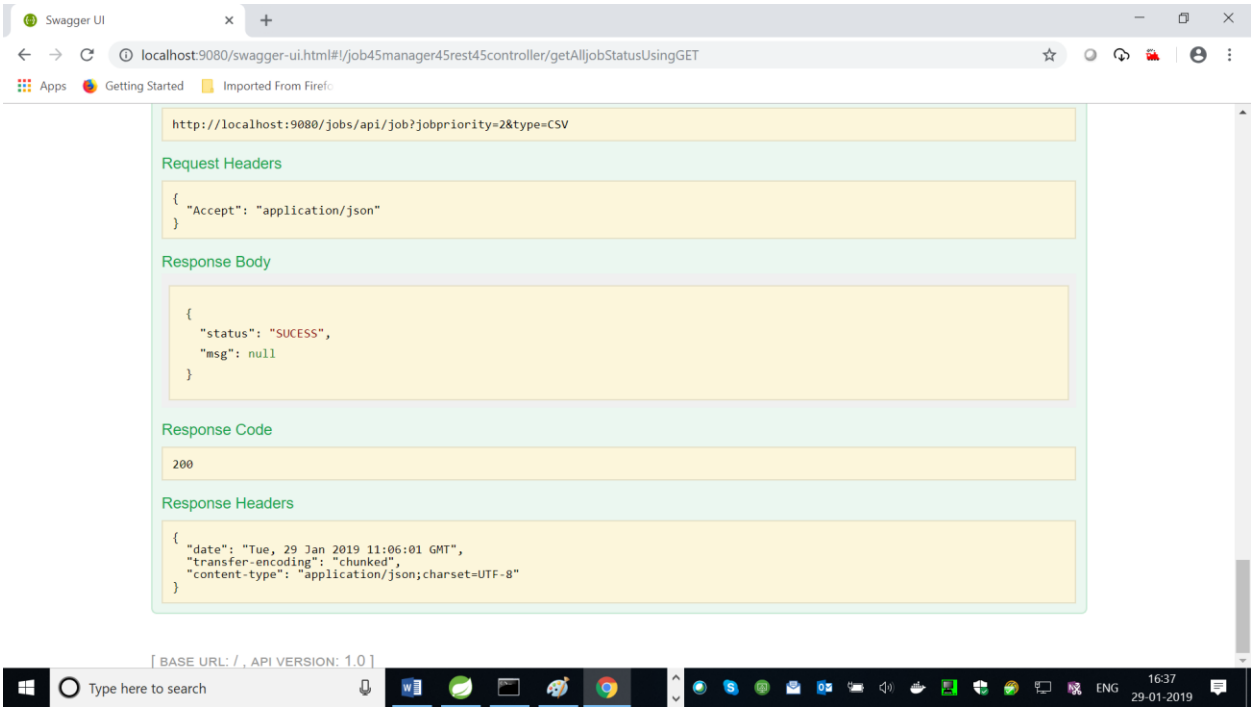


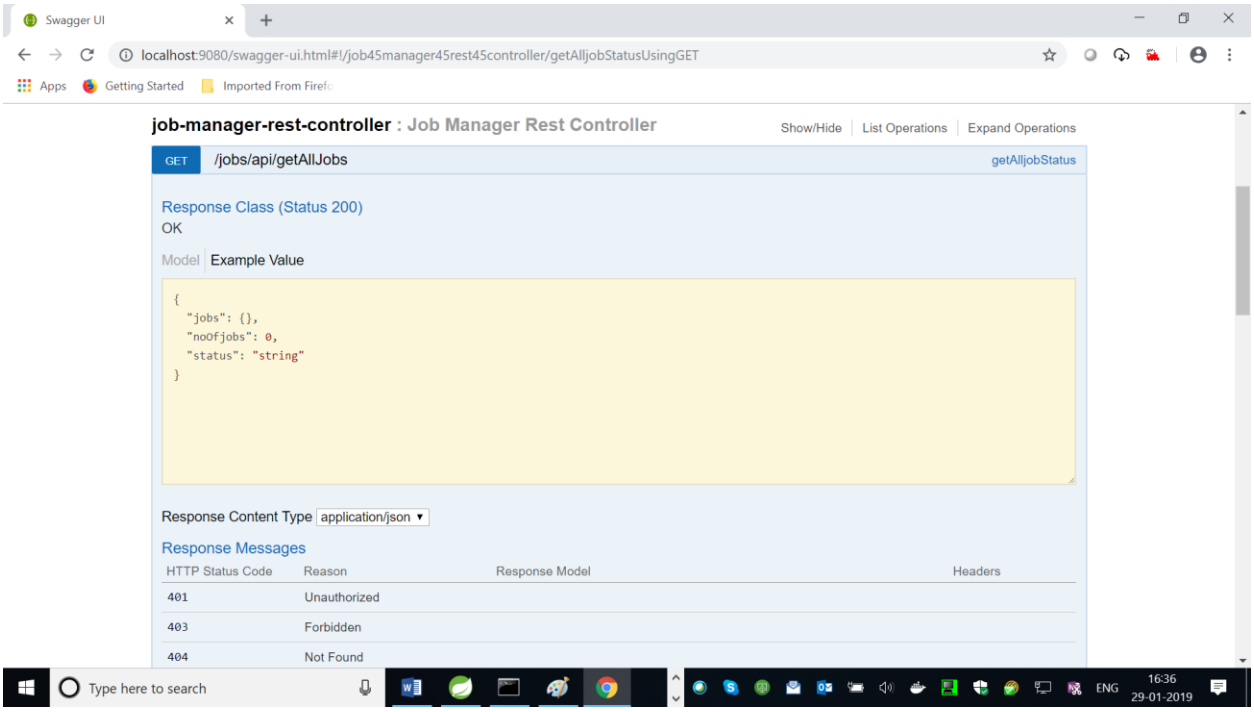
Click on SubmitJobType



Valid JOB type are email, CSV, file etc. which are sample Job Implementation

Click on “Try it Out!”





Sample Response


```
{
  "noOfjobs": 6,
  "status": "SUCESS",
  "jobs": {
    "CSV@1548818995151": {
      "status": "SUCCESS",
      "name": "CSV@1548818995151",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:55.153+0000",
      "priority": 2,
      "submitedTime": null,
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    },
    "CSV@1548818995712": {
```

```
      "status": "SUCCESS",
      "name": "CSV@1548818995712",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:55.713+0000",
      "priority": 2,
      "submitedTime": null,
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    },
    "CSV@1548818995931": {
      "status": "QUEUED",
      "name": "CSV@1548818995931",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:55.931+0000",
      "priority": 2,
      "submitedTime": null,
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    },
    "CSV@1548818995554": {
      "status": "SUCCESS",
      "name": "CSV@1548818995554",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:55.554+0000",
      "priority": 2,
      "submitedTime": null,
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    },
    "CSV@1548818995377": {
      "status": "SUCCESS",
      "name": "CSV@1548818995377",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:55.379+0000",
      "priority": 2,
      "submitedTime": null,
```

```
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    },
    "CSV@1548818996250": {
      "status": "SUCCESS",
      "name": "CSV@1548818996250",
      "timeStarted": "2019-01-29T18:30:00.000+0000",
      "scheduledStartTime": "2019-01-30T03:29:56.251+0000",
      "priority": 2,
      "submitedTime": null,
      "completedTime": null,
      "configFile":
"C:\\Users\\DineshKori\\githibfinal\\optile\\JobImplementor\\src\\main\\java\\com\\optile\
\jobs\\helper\\SampleCSVHelper1.properties",
      "startTime": "2019-01-29T18:30:00.000+0000",
      "jobRollable": false,
      "immediateJob": false,
      "configureable": true,
      "cpuintensiveJob": false
    }
  }
}
```

12. FUTURE ENHANCEMENTS

1. Persist all the Job in the DB for Reporting purpose
2. Simple Dashboard UI could be provided to get the status of Job, Queued job, Failed Job etc.