User data management project is divided into 3 parts.

1. dataLoaderApp (Play Framework & AngularJs 1)
2. reactiveJobQueue (Scala, Kafka)
3. workerApp (Scala)

Requirement

1. SBT
2. Kafka
3. FTP server (I used https://github.com/stilliard/docker-pure-ftpd)
4. MYSql (https://github.com/mysql/mysql-docker)

Configuration

Each project have application.conf file, we can set FTP, MYSql details their.

Create kafka topic

bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic uploadmessage

Create MySql database & Table.

Create database userdb;
Use userdb;
create table userinfo(id varchar(255) primary key,name varchar(255), time_of_start varchar(255));

Running Application :

1. dataLoaderApp : (Prod mode)

```
$ cd dataLoaderApp
$ sbt dist
$ unzip dataloaderApp/target/universal/dataloaderapp-1.0-SNAPSHOT.zip
$ cd dataloaderapp-1.0-SNAPSHOT
$ ./bin dataloaderapp
```

OR

```
$ cd dataLoaderApp
$ sbt run
```

2. reactiveJobQueue
This application supports workerApp, First need to create jar and publish to local .ivy2 folder.

```
$ cd reactiveJobQueue
$ sbt clean publishLocal
```

```
libraryDependencies += "org.rahul" % "reactivejobqueue_2.11" % "1.0"
```

Check build.sbt, adjust following if require.

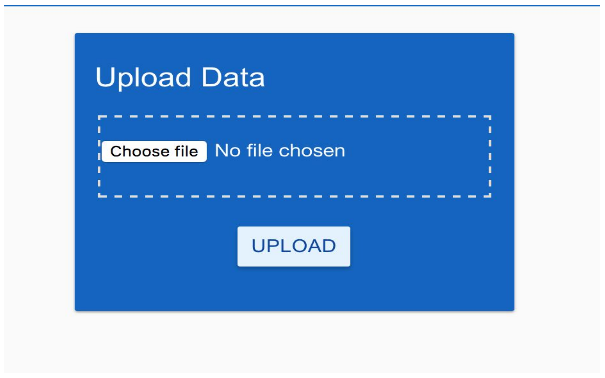
```
resolvers += Resolver.file("Local", file( Path.userHome.absolutePath +
"/.ivy2/local"))(Resolver.ivyStylePatterns)
```

3. workerApp

```
$ sbt run
```

Workflow

Step 1: User Upload CSV data on Server, Data will store on server's **/tmp** folder location.

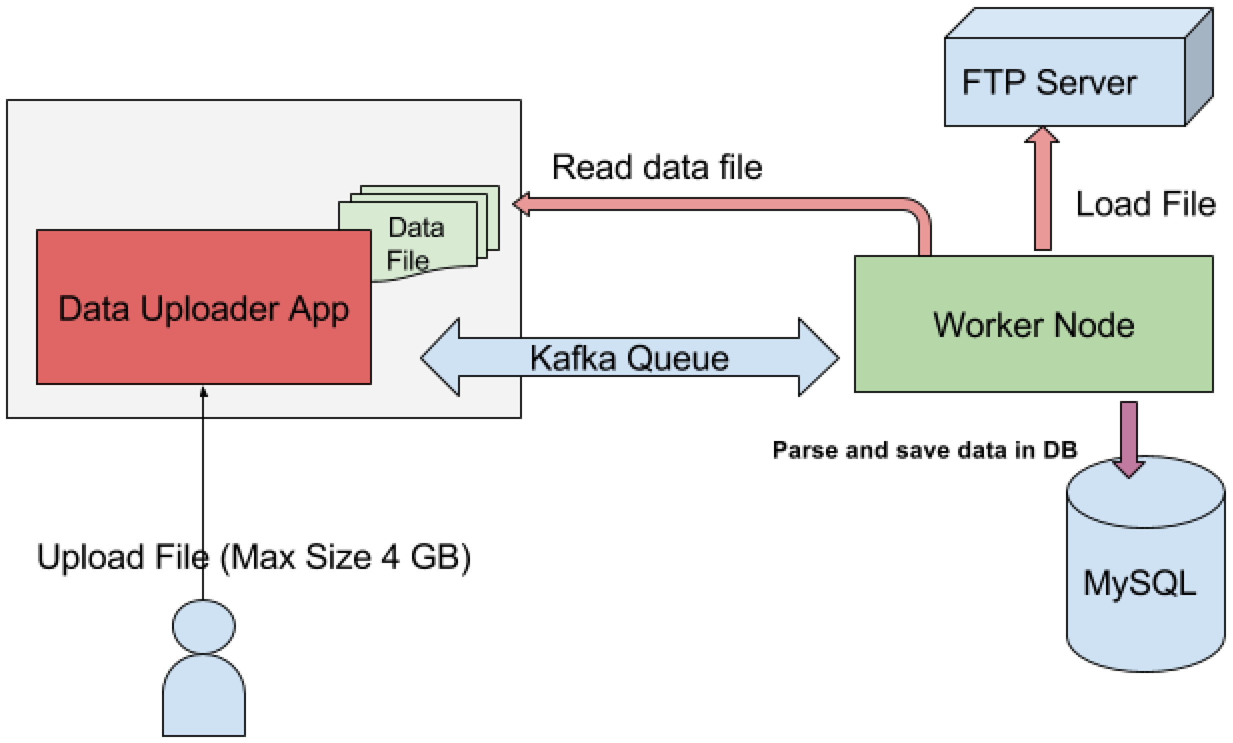


Step 2: Web server will notify worker by sending message on Kafka channel.

```
(65dd53e8-c5aa-4c9d-9583-6a212196e32f,
{"uuid":"65dd53e8-c5aa-4c9d-9583-6a212196e32f","filename":"data_test.csv","size":118036394
2,"startTime":1488845854788,"endTime":1488845854801,"totalUploadTime":13,"path":"/tmp/dat
a_test.csv","filetype":"text/csv"})
```

Step 3: Worker node will parse message and read data from web servers /tmp location to FTP server.

Step 4: After FTP Upload , worker will run DBSroreService.
DBService is doing small wrangler job like changing data to UTC, name to smaller case.
While storing in DB Only unique ID stored in DB.



Missing Items

Test cases are missing, not able to give much time due to on going project at office.
No Metadata stores implementation are present for file upload and db process failover.

Improvement

Worker node will move on Apache Spark for distributed file upload and processing on scale.