**[Assignment]**

**Application (Web) Technical Specification**

Submitted By

[Manish Yadav]

Contents

[1 Introduction 2](#_Toc497001000)

[1.1 Project Objectives 2](#_Toc497001001)

[2 Design Guidelines and Approach 3](#_Toc497001002)

[2.1 Assumptions / Constraints / Standards 3](#_Toc497001003)

[2.1.1 Functional assumptions 3](#_Toc497001004)

[3 Architecture 3](#_Toc497001005)

[3.1 Servers 3](#_Toc497001006)

[3.1.1 System requirements for deployment 3](#_Toc497001007)

[3.2 Deployment and Configuration 4](#_Toc497001008)

# Introduction

As a part of a scrum team developing data warehouse for Bloomberg to analyze FX deals. One of customer stories is to import deals details from files into DB.

## Project Objectives

The tool should allow user to upload .csv file containing deals data and should be able to import the data with a performance of 100,000 records in less than 5 seconds.

# Design Guidelines and Approach

* File format should be .CSV and should contain the following fields (Deal Unique Id, From Currency ISO Code "Ordering Currency", To Currency ISO Code, Deal timestamp, Deal Amount in ordering currency).
* Row structure should be validated.
* Valid rows should be stored in table/document, with reference to source file name .
* Invalid rows should be stored in another table/document, with reference to source file name.
* The DB contains a third table to maintain accumulative count of deals per Ordering Currency "Columns : Currency ISO Code, CountOfDeals ". Upon completion of importing process, the system should increase count of deals per currency.
* System should not import same file twice.
* No rollback is allowed, all imported rows should be saved in DB.

## Assumptions / Constraints / Standards

### Functional assumptions

* Order of columns is same as given in requirement specification. There is no exchange in columns (order should be same otherwise data importing will be inaccurate)
* **Deal Unique Id** in file should be unique (otherwise there will be duplicate results)
* **Timestamp** should should be in the format DD/MM/YYYY HH:MM:SS

# Architecture

The tool can be broadly classified into two parts.

* UI sends requests to backend for fetching data.
* Service wich communicates with database and renders results back to UI.

## Servers

Apache Tomcat Tested on (Apache Tomcat 7.0.8)

### System requirements for deployment

* Java Version 1.8
* Apache tomcat
* Maven
* Mysql Database 5.x
* Browser(Firefox Chrome)

## Deployment and Configuration

The deployment steps are as follows :

1. System shuold have apache tomcat installed, and in running state.
2. MySql database Server should be installed and running.
3. First create database and tables by using following queries.

//Create Databse

1. Create database fx\_deals

//Use above database

1. Create table “valid\_deal”

**CREATE** **TABLE** `valid\_deal` (

`DEAL\_ID` **varchar**(100) **NOT** **NULL**,

`FROM\_CURRENCY\_ISO\_CODE` **varchar**(100) **DEFAULT** **NULL**,

`TO\_CURRENCY\_ISO\_CODE` **varchar**(100) **DEFAULT** **NULL**,

`TIME\_STAMP` **varchar**(100) **DEFAULT** **NULL**,

`AMOUT\_ORDERING\_CURRENCY` **varchar**(100) **DEFAULT** **NULL**,

`FILE\_ID` **varchar**(100) **NOT** **NULL**

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8;

1. Create Second table “invalid\_deal”

**CREATE** **TABLE** `invalid\_deal` (

`DEAL\_ID` **varchar**(100) **NOT** **NULL**,

`FROM\_CURRENCY\_ISO\_CODE` **varchar**(100) **DEFAULT** **NULL**,

`TO\_CURRENCY\_ISO\_CODE` **varchar**(100) **DEFAULT** **NULL**,

`TIME\_STAMP` **varchar**(100) **DEFAULT** **NULL**,

`AMOUT\_ORDERING\_CURRENCY` **varchar**(100) **DEFAULT** **NULL**,

`FILE\_ID` **varchar**(100) **NOT** **NULL**

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8;

1. Create Third Table “deal\_file”

**CREATE** **TABLE** `dealfile` (

`ID` **varchar**(100) **NOT** **NULL**,

`FILE\_NAME` **varchar**(100) **DEFAULT** **NULL**,

**PRIMARY** **KEY** (`ID`)

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8;

1. Create Fourth table “deal\_accumulative\_count”

**CREATE** **TABLE** `deal\_accumulative\_count` (

`Ordering\_Currency` **varchar**(100) **NOT** **NULL**,

`Ordering\_Currency\_COUNT` **bigint**(20) **DEFAULT** **NULL**

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8;

1. One important thing set Global Max Packet (don’t miss it)

SET GLOBAL max\_allowed\_packet=1073741824;

You can use same username and password for Application:

Username : root

Passowrd: root

1. Next Step to deploy the war file.
2. Download the war file(**assignment-ui.war** and **assignment-service.war** ) from following location

!**important :please make changes in database.propeties file to make the connection**

**For example :**

# JDBC For my sql

jdbc.driver=com.mysql.jdbc.Driver

jdbc.URL=jdbc:mysql://localhost:3306/fx\_deals

jdbc.user=root

jdbc.password=root

jdbc.cache.InitialLimit=1

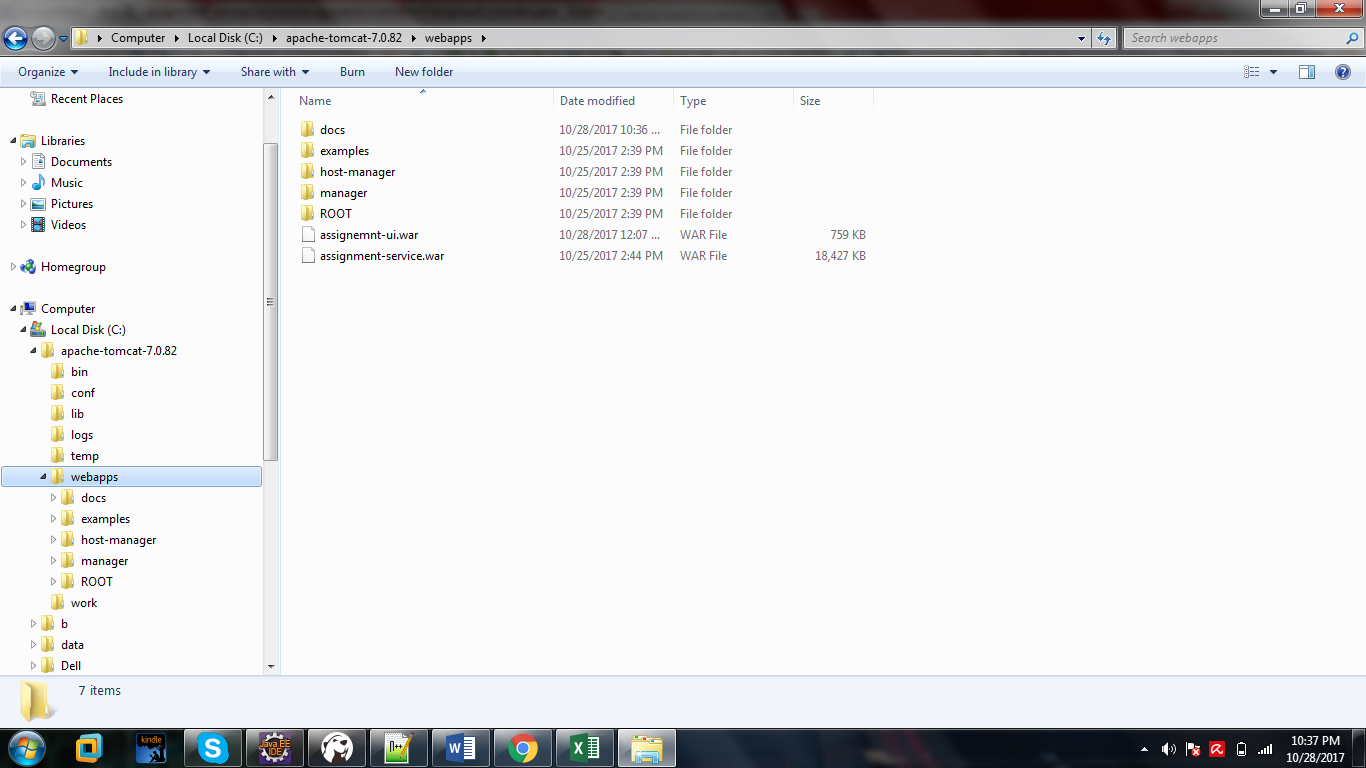
jdbc.cache.MinLimit=1

jdbc.cache.MaxLimit=20

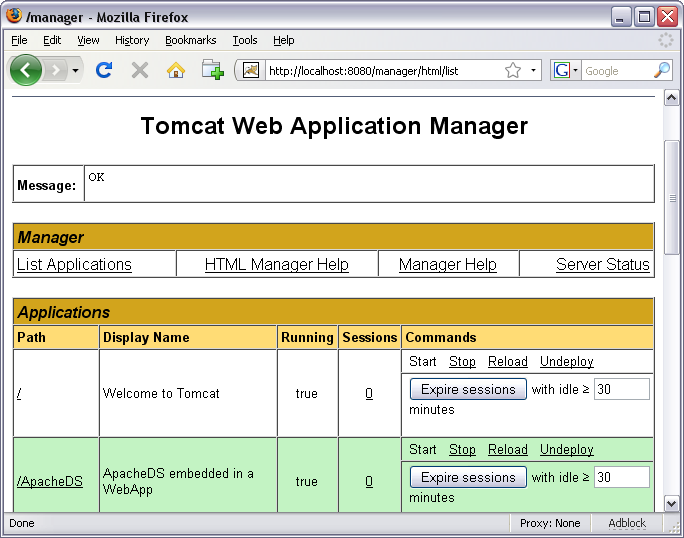
Or just do maven install to create the war file for deployment

1. Copy and paste the war file in tomcat apache Webapps folder and restart the server

for example : if C:\apache-tomcat-7.0.82\webapps is the path of installation directory, just paste the war files here.



1. Or you can deploy by tomcat manager screen.



After deployment, access the following Url

http://<ip or machine name >:<port>/ assignemnt-ui/#/

for Example

<http://localhost:8080/assignemnt-ui/#/>

You should see the screen below

