

Technology Management System

Java/Jee Web Applciation

Vijaykumar Vishwakarma | 11/30/2016

Contents

[Introduction 2](#_Toc468284803)

[Scope 2](#_Toc468284804)

[Technology Used/Required 2](#_Toc468284805)

[Installation 3](#_Toc468284806)

[Architecture 4](#_Toc468284807)

[Design Models 6](#_Toc468284808)

[Use Cases 6](#_Toc468284809)

[Class Diagram 7](#_Toc468284810)

[Testing 8](#_Toc468284811)

# Introduction

Technology Management System(TMS) is a web application that manages the list of technologies available in market. With TMS application the user can add, modify, delete and list the technologies. TMS is a single page application.

TMS exposes RESTful services for adding, modifying, deleting and listing the technologies.

# Scope

This document provides brief information on installing and running the TMS application, technologies used, architecture and testing the application

# Technology Used/Required

* Java 7
* Spring 4.1
* Spring JDBC
* Spring MVC
* MySQL 5
* Angular JS 1.4
* Servlet
* RESTful Web Services
* Jackson
* JSON
* AJAX
* Apache Commons DBCP
* Apache Tomcat
* Maven 3
* Junit 4
* HTML
* CSS
* Mozilla Firefox with Http Requester Add On
* Git
* Eclipse Neon

# Installation

Following are the steps to install and run the application

* Clone project from Vijay’s Github account using the URL OR explode the attached zip file.
* Change directory to TechnologyManagement folder.
* Run maven command to clean, just to be sure clean target directory

**mvn clean**

* Run maven command to package the deployable war file.

**mvn package**

* Run the techmgmt.sql file to create database table and insert sample records

SQL

* Update the techmgmt.properties file with your specific database properties.
* Deploy the created war file in to application/web server. I used tomcat to deploy application.
* Start the application/web server.
* Access url

http://<machine or ip address>:<portno>/TechnologyManagement/techView.html

# Architecture

**Technology Management System**

AJAX calls

View (Angular JS, HTML, CSS)

Controller (Spring REST, Servlet)

Service (Spring )

DAO (Spring JDBC)

TECH (MySQL)

TMS is developed using MVC architecture.

**View** is the HTML page which is developed using Angular JS, HTML, CSS, JSON

**Controller** is a Spring RESTful controller which defines and implements add, update, delete and list services. These services consumes/produces JSON data as a result. Data binding is done using Jackson.

Rest Controller gets the data by invoking the methods on TechnologyService class which is injected and autowired by type.

**Model** comprises of Service and DAO classes. Service works as Façade to the controller. Service get the data by making calls to the TechnologyDAO class which is injected and autowired by type.

DAO class uses JdbcTemplate of Spring JDBC to execute database queries. TechnologyRowMapper utility class is used to bind the database values to the enity property.

# Design Models

## Use Cases

## Class Diagram

* Technology Entity (domain object)

**Technology**

- id : int

- name : String

- company : String

- year: int

* TechnologyService

**TechnologyService**

- technologyDAO : TechnologyDAO

+ getTechnology(id : int) : Technology

+ deleteTechnology(id : int) : void

+ addTechnology(e : Technology) : Technology

+ updateTechnology(e : Technology) : Technology

+ getAllTechnologies() : List<Technology>

* TechnologyDAO

**TechnologyDAO**

- jdbcTemplate : JdbcTemplate

+ getTechnology(id : int) : Technology

+ deleteTechnology(id : int) : void

+ addTechnology(e : Technology) : Technology

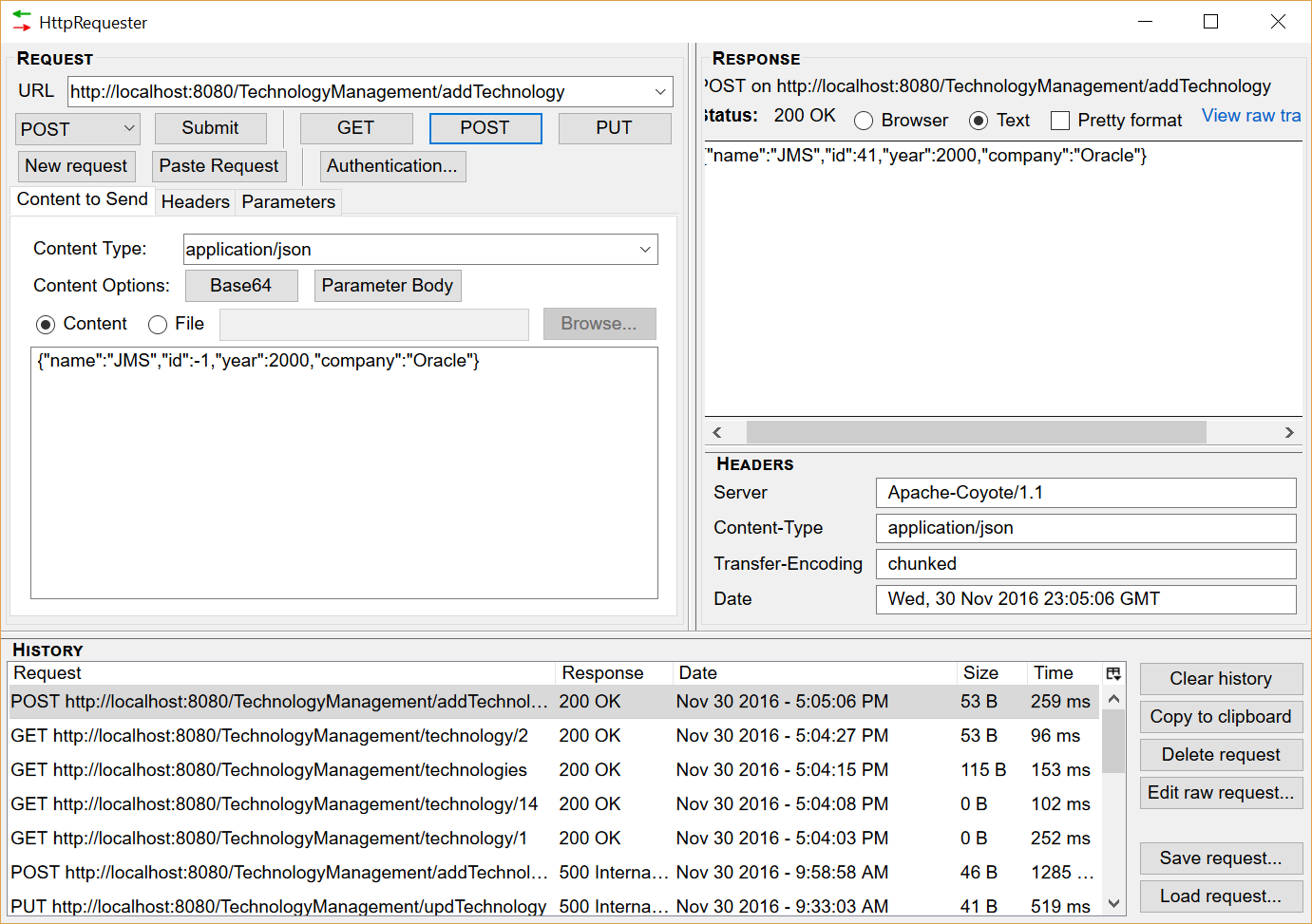
+ updateTechnology(e : Technology) : Technology

+ getAllTechnologies() : List<Technology>

# Testing

* When mvn package command is executed Junit Test cases run automatically.
* To test REST services Http Requester Firefox Add on is used

Following is sample screen for testing using Http Requester



**Test data**

**ADD**

Url : <http://localhost:8080/TechnologyManagement/addTechnology>

Method: POST

Content-Type : application/json

Body : {"name":"JMS","id":-1,"year":2000,"company":"Oracle"}

**UPDATE**

Url : <http://localhost:8080/TechnologyManagement/updTechnology>

Method: PUT

Content-Type : application/json

Body : {"name":"JMS1","id":41,"year":2000,"company":"Oracle"}

**DELETE**

Url : <http://localhost:8080/TechnologyManagement/delTechnology/41>

Method: DELETE

**LIST**

Url: <http://localhost:8080/TechnologyManagement/technologies>

Method: GET