

3-2017

AUTOMATIC GENERATION OF WEB APPLICATIONS AND MANAGEMENT SYSTEM

Yu Zhou
004306704@coyote.csusb.edu

Follow this and additional works at: <https://scholarworks.lib.csusb.edu/etd>



Part of the [Computer and Systems Architecture Commons](#), and the [Data Storage Systems Commons](#)

Recommended Citation

Zhou, Yu, "AUTOMATIC GENERATION OF WEB APPLICATIONS AND MANAGEMENT SYSTEM" (2017).
Electronic Theses, Projects, and Dissertations. 434.
<https://scholarworks.lib.csusb.edu/etd/434>

This Project is brought to you for free and open access by the Office of Graduate Studies at CSUSB ScholarWorks. It has been accepted for inclusion in Electronic Theses, Projects, and Dissertations by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.

AUTOMATIC GENERATION OF WEB APPLICATIONS AND
MANAGEMENT SYSTEM

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Yu Zhou
March 2017

AUTOMATIC GENERATION OF WEB APPLICATIONS AND
MANAGEMENT SYSTEM

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Yu Zhou
March 2017
Approved by:

Dr. Zhengping Wu, Advisor, Computer Science and Engineering

Dr. George Georgiou, Committee Member

Dr. Owen J. Murphy, Committee Member

© 2017 Yu Zhou

ABSTRACT

One of the major difficulties in web application design is the tediousness of constructing new web pages from scratch. For traditional web application projects, the web application designers usually design and implement web application projects step by step, in detail. My project is called “automatic generation of web applications and management system.” This web application generator can generate the generic and customized web applications based on software engineering theories. The flow driven methodology will be used to drive the project by Business Process Model Notation (BPMN). Modules of the project are: database, web server, HTML page, functionality, financial analysis model, customer, and BPMN. The BPMN is the most important section of this entire project, due to the BPMN flow engine that most of the work and data flow depends on the engine. There are two ways to deal with the project. One way is to go to the main page, then to choose one web app template, and click the generating button. The other way is for the customers to request special orders. The project then will give suitable software development methodologies to follow up. After a software development life cycle, customers will receive their required product.

TABLE OF CONTENTS

ABSTRACT	iii
LIST OF FIGURES	vi
CHAPTER ONE INTRODUCTION	1
CHAPTER TWO DEVELOPMENT TOOLS AND ENVIRONMENTS.....	2
CHAPTER THREE IMPLEMENTATION	
Implement Plan.....	4
Libraries and Techniques.....	6
Maven	6
MVC Framework	7
Spring Framework.....	8
Hibernate Framework.....	9
BPMN.....	10
SOAP	12
REST	13
jQuery Ajax	14
CHAPTER FOUR PROJECT CONSTRUCTION	
Use Case Diagrams.....	15
Main ER Diagrams.....	20
CHAPTER FIVE PROJECT DATABASE.....	23
CHAPTER SIX PROJECT COMPONENT	
System Configuration.....	33
Setup Customer Model	37

Setup Tomcat Web Service	39
Creating Project	45
Deploying Project and Start Web Service	47
Deploying BPMN Process Flow	48
Assign Tasks	52
Send Message	58
CHAPTER SEVEN CONCLUSION	59
APPENDIX CODE OF CRITICAL PARTS	61
REFERENCES	105

LIST OF FIGURES

Figure 3.1. MVC.	7
Figure 3.2. Table of country.....	10
Figure 3.3. jQuery ajax work flow.	14
Figure 4.1. Overview of system construction.....	15
Figure 4.2. System configuration component use case.	16
Figure 4.6. Project component use case.	18
Figure 4.7. Task and database component use case.....	19
Figure 4.8. Customer ER component.	20
Figure 4.9. Database component.	21
Figure 4.10. Project ER component.....	21
Figure 4.11. Tomcat ER component.....	22
Figure 5.1. Table of country.....	23
Figure 5.2. Table of state.....	23
Figure 5.3. Table of department.	23
Figure 5.4. Table of unit.....	23
Figure 5.5. Table of authorizes.	24
Figure 5.6. Table of variable.	24
Figure 5.7. Table of role.	24
Figure 5.8. Table of menu.	24
Figure 5.9. Table of task type.	25
Figure 4.10. Table of page type.....	25

Figure 5.11. Table of system port.....	25
Figure 5.12. Table of system variable configuration.....	25
Figure 5.13. Table of customer.....	26
Figure 5.14. Table of page type.....	26
Figure 5.15. Table of task.....	27
Figure 5.16. Table of project.....	27
Figure 5.17. Table of project file.....	28
Figure 5.18. Table of customer.....	28
Figure 5.19. Table of upload BPMN file.....	29
Figure 5.20. Table of database.....	29
Figure 5.21. Table of database table.....	29
Figure 5.22. Table of database table variable.....	29
Figure 5.23. Table of page.....	30
Figure 5.24. Table of message.....	30
Figure 5.25. Table of tomcat config.....	31
Figure 5.26. Table of file.....	31
Figure 5.27. Table of tomcat.....	32
Figure 6.1. System variable configuration.....	34
Figure 6.2. System port configuration.....	34
Figure 6.3. Variable type configuration.....	35
Figure 6.4. Page type configuration.....	36
Figure 6.5. Task type configuration.....	36

Figure 6.6. State configuration.....	37
Figure 6.7. Country configuration.	37
Figure 6.8. Company configuration.....	38
Figure 6.9. Role configuration.....	38
Figure 6.10. Department configuration.	39
Figure 6.11. Country configuration.	39
Figure 6.12. Tomcat web service list.	40
Figure 6.13. Add tomcat web service page.	40
Figure 6.14. Tomcat configuration list.....	40
Figure 6.15. Add tomcat web service page.	41
Figure 6.16. Tomcat configuration list.....	42
Figure 6.17. Tomcat file content.	42
Figure 6.18. Add tomcat folder.	43
Figure 6.19. Add tomcat file.....	43
Figure 6.19. Tomcat web service console.	44
Figure 6.20. Project list.....	45
Figure 6.21. Add project page.	45
Figure 6.22 Two ways to add project files.	46
Figure 6.23. Tomcat web service options.	47
Figure 6.24. Project options.....	47
Figure 6.25. Created on system side BPMN process list.....	48
Figure 6.26. Uploaded BPMN process file list.	49

Figure 6.27. Deployed uploaded BPMN process list.	49
Figure 6.28. BPMNN process file in xml format.	50
Figure 6.29. BPMNN process displayed by image.	51
Figure 6.30. Embedded BPMN process editor.	51
Figure 6.31. Current BPMN process task list.....	52
Figure 6.32. In process BPMN task list.....	52
Figure 6.33. Finished BPMN task list.....	53
Figure 6.34. Add BPMN task page.	54
Figure 6.35. Add generic task page.....	55
Figure 6.36. Received task list.....	55
Figure 6.37. System tool bar task list.....	56
Figure 6.38. To do list of task.	56
Figure 6.39. Finish a task.	57
Figure 6.40. Message list.	58
Figure 6.41. System tool bar message list.....	58

CHAPTER ONE

INTRODUCTION

One of the major difficulties in web application design is the tediousness of constructing new web pages from scratch. For traditional web application projects, customers usually need to find software companies who can design web application and follow software development methodologies to design customers' web applications in the past few years. Recently, many online web application generators have appeared. Weebly and WordPress (online web application builder tool) have become the most used online web project generators in the U. S., but their customers still need to modify their projects such as re-layout pages and add additional elements to their web applications. The aim of the project is to make it more user-friendly for customers, who not familiar the general technology of web application. This project is named "Automatic Generation of the Web Applications." The project will use flow driven methodology to generate web applications. Based on the flow driven methodology, Business Process Model Notation (BPMN) is the top choice. The project will also include the software engineering theories because some ways of generating web applications are for customers to submit their special requirements on customized project order pages. The project will use suitable software development methodologies to match customers' web application projects. After a software development life cycle, customers will receive their required product and then they need to add them to their projects.

CHAPTER TWO

DEVELOPMENT TOOLS AND ENVIRONMENTS

There are four main development tools that I used them to design and develop my project and web service. First, Eclipse is used to develop the project and web service. Second, Apache Tomcat is used as a web host environment for the project and web service. Third, MySQL is used as a data store where data will be saved into. The last one is UML editor that will be used to model a relationship between database and project components.

Eclipse, which is an integrated development environment (IDE) and what is the most widely used Java IDE, offers a base workspace and an extensible plug-in component for customizing development environment [19]. Because of the Windows-based computer, third party plug-ins can be easily embedded into Eclipse development environment and assistant Java web developers to develop project such as SVN. The eclipse also provides powerful support for development and testing of the project and web services, containing the capability of debugging, inspect variables and step through the code.

Apache Tomcat is an open-source and widely used Java Servlet Container developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, Java Server Pages (JSP). On Eclipse platform, the tomcat web service can be controlled by the Eclipse IDE [20].

MySQL is an open-source relational database management system (RDBMS). Eclipse provides many database interfaces including MySQL and it is easy to setup on the Windows-based environment to produce the project database by using Eclipse database tool [21].

In the field of software engineering, the UML stands for Unified Modeling Language, which is a general-purpose, developmental and modeling language. The UML is intended to provide a standard way to visualize the design of a project here [22].

CHAPTER THREE

IMPLEMENTATION

Implement Plan

To design the automatic generation of web application and management system project, I will use the spiral software development model, which is a risk-driven software process framework [7]. At the first development life cycle, I will finish 40% functionalities of the total project. The 40% functionalities of the whole project will include the BPMN, which is a systemic approach for capturing, designing, executing, documenting, measuring, monitoring, and controlling both automated and non-automated processes to meet the objectives and business strategies of a company [3].

The BPMN can offer most of the flexible workflows and three static logic flows. Due to the BPMN technology, the workflow can be generated and modified by BPMN flow tool. In my project, I will use this technology for the web page workflow. For logic flows, I will use the flow controller to control these three static logic flows. They are online shopping logic flow, online hotel logic flow, and online restaurant reservation flow. According to the BPMN's workflow, the flow controller can decide what page and what page's data should be pop out when customers make requirements on that web page. Customers can modify the workflow at Customer's management platform. According to the technology I decide to use, the project's construction is also important. All web pages and

functions will be decomposed into code fragments and save into database. Html and jQuery will make the web pages more flexible and user-friendly.

BPMN can also guide the customers to finish their first web application, web application data controller (where customers and system administrators can control and manage data at different interfaces), Email controller as a notification functionality that will be embedded into web applications, web server controller (where customers and system administrators can control the tomcat at different interfaces), administration platform (where system administrators can manage the system's configuration, web application's template, functionality, and database.) The Database controller can be also used and managed by customers at customer's interfaces (where customers only can see the data and manage the data rather than the database itself). To design this project, I use Java 7.0 as the basic development language, BPMN as work and logic flow controller, Html [6], jQuery [9], Tomcat [5], Apache [2], Java SSH (Spring, Hibernate, and Struts) framework [1][8] and Maven together [10]. On web application page, the web application generator can offer these elements that they are text boxes, area text box, submit button, reset button, customized button, search area, single image and slide image box, text label, sub window, and section area.

Libraries and Techniques

Maven

I chose to use Maven for my project. That was because the Wikipedia said “Maven was originally started as an attempt to simplify the build processes in the Jakarta Turbine project. There were several projects each with their own Ant build files that were all slightly different and JARs were checked into CVS [17]”. The primary goal of Maven is to accept the Java developers to comprehend the whole state of the development effort with a short period [17]. The Wikipedia also said that “in order to attain this goal there are several areas of concern that Maven attempts to deal with:

- Making the build process easy.

- Providing a uniform build system.

- Providing quality project information.

- Providing guidelines for best practices development.

- Allowing transparent migration to new features [17].”

The Maven was used as a project construction management tool to implement the project because the Maven can offer me an overview of my project construction such as collection of java jars and embedded plug-ins tomcat. If I choose not use the Maven, then I should upload and import library references to the project. If I change the project workspace path, I must re-point the library references to the project.

MVC Framework

Model–view–controller (Figure 3.1) is a software design pattern for implementing customer interfaces on computers. It divides a given software application into three interconnected parts [18]. To connect each component between model and controller, spring framework and java annotation was used to connect them. When a model is declared, then I should add a java entity annotation and put it above a class. To separate internal representations of information from the ways that information is presented to or accepted from the customers. The model directly manages the data, logic, and rules of the application. The view can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants. The third part, the controller, accepts input and converts it to commands for the model or view [18].

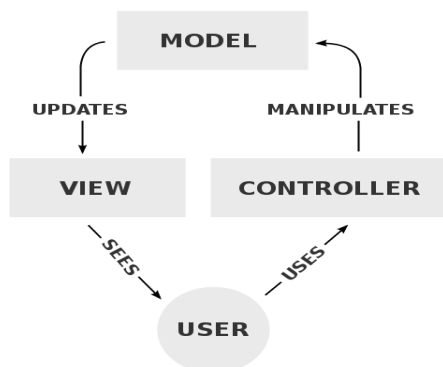


Figure 3.1. MVC.

Spring Framework

The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform [11]. Although the spring framework does not include any specific programming model, it has become popular and been using widely in the Java community.

The spring framework can provide a comprehensive programming and configuration model for my Java-based web applications. Based on this benefit, the spring framework allows me to embedded other components easily such as Hibernate, BPMN engine, MVC design pattern. To embed another component into spring, I need to make other components as objects and then inject them into spring XML file.

Hibernate Framework

Hibernate ORM (hibernate in short) is an object-relational mapping tool for the Java language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions [14].

The main hibernate's feature is mapping from Java classes to database tables, and mapping from Java data types to database data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual handling and object conversion of the result set.

Using the hibernate framework to manage database is easier than using direct database connection method because the hibernate framework will help me to map java entity elements with database table elements by add annotation at each entity class and inject them into hibernate configuration file. If I use direct database connection method to implement the project, I must implement the database connection method and connection pool, execute queue, and map data manually.

BPMN

Business Process Model and Notation (BPMN) is a graphical representation for specifying business processes in a business process model.

Business Process Management Initiative (Figure 3.2) developed BPMN, which has been maintained by the Object Management Group since the two organizations merged in 2005. Version 2.0 of BPMN was released in January 2011, at which point the name was adapted to Business Process Model and Notation as execution semantics were also introduced alongside the notational and diagramming elements [16].

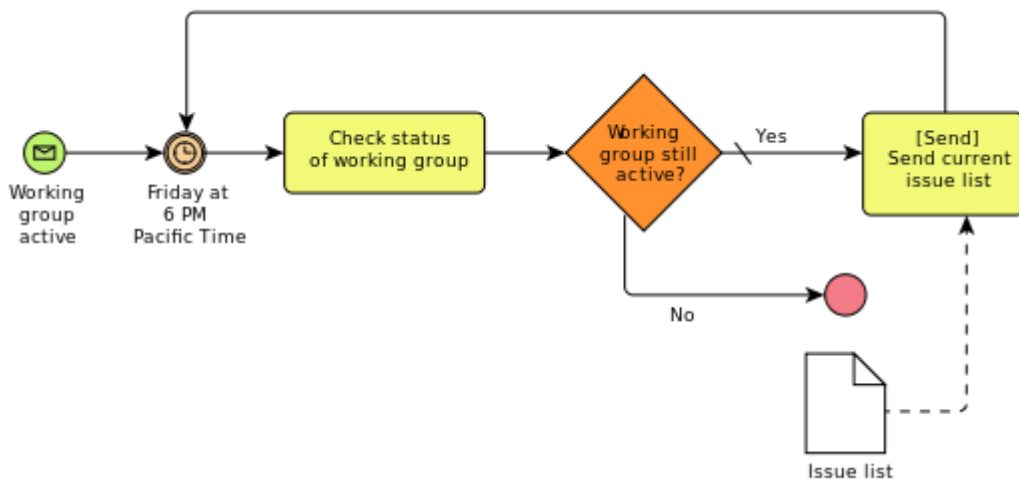


Figure 3.2. Table of country.

Because BPMN can be embedded into spring framework, the whole workflow logic is not needed when implementing workflow component. What I

need to do is to design a BPMN process chart and some extra java function or js functions that the process may use.

SOAP

SOAP (Simple Object Access Protocol) is a protocol specification for exchanging structured information in the implementation of web services in computer networks. Its purpose is to induce extensibility, neutrality, and independence. It uses XML Information Set for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission. SOAP allows processes running on disparate operating systems (such as Windows and Linux) to communicate using Extensible Markup Language (XML). Since Web protocols like HTTP are installed and running on all Operating systems, SOAP allows clients to invoke web services and receive responses independent of language and platforms [12].

This project has a subsystem to work with the main system. That system is a file system which handles all files and folder operation, including creation, duplication, deletion, and update. The main system will send a command with parameters to the subsystem. If it need to operate files such as copying original tomcat files, updating tomcat files, and uploading file from the customer system to the service. Eclipse was used to add web service and web client to these systems. All transactions are byte data type and in JSON style.

REST

Representational State Transfer (REST) or RESTful Web services are one way of providing interoperability between computer systems on the Internet. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations [13]. Using HTTP, the kind of operations available include those predefined by the HTTP verbs GET, POST, PUT, DELETE and so on.

REST is used as the connection between the main system and BPMN engine. The REST has less work to do than the SOAP, and the REST will return a JSON string to the client and display the content on the customer screen. The SOAP should have a client side as a message receiver, but the REST does not need. So, the REST can save more work than developing a SOAP connection.

jQuery Ajax

The Wikipedia recommended me to use the jQuery for my project front side. The reason is “jQuery (Figure 3.3) is a cross-platform JavaScript library designed to simplify the client-side scripting of HTML. jQuery is the most popular JavaScript library in use today, with installation on 65% of the top 10 million highest-trafficked sites on the Web. jQuery is free, open-source software licensed under the MIT License [15].”

Ajax is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications. With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

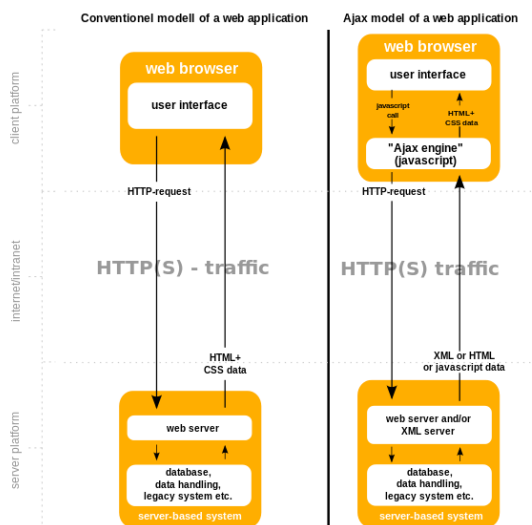


Figure 3.3. jQuery ajax work flow.

CHAPTER FOUR

PROJECT CONSTRUCTION

Use Case Diagrams

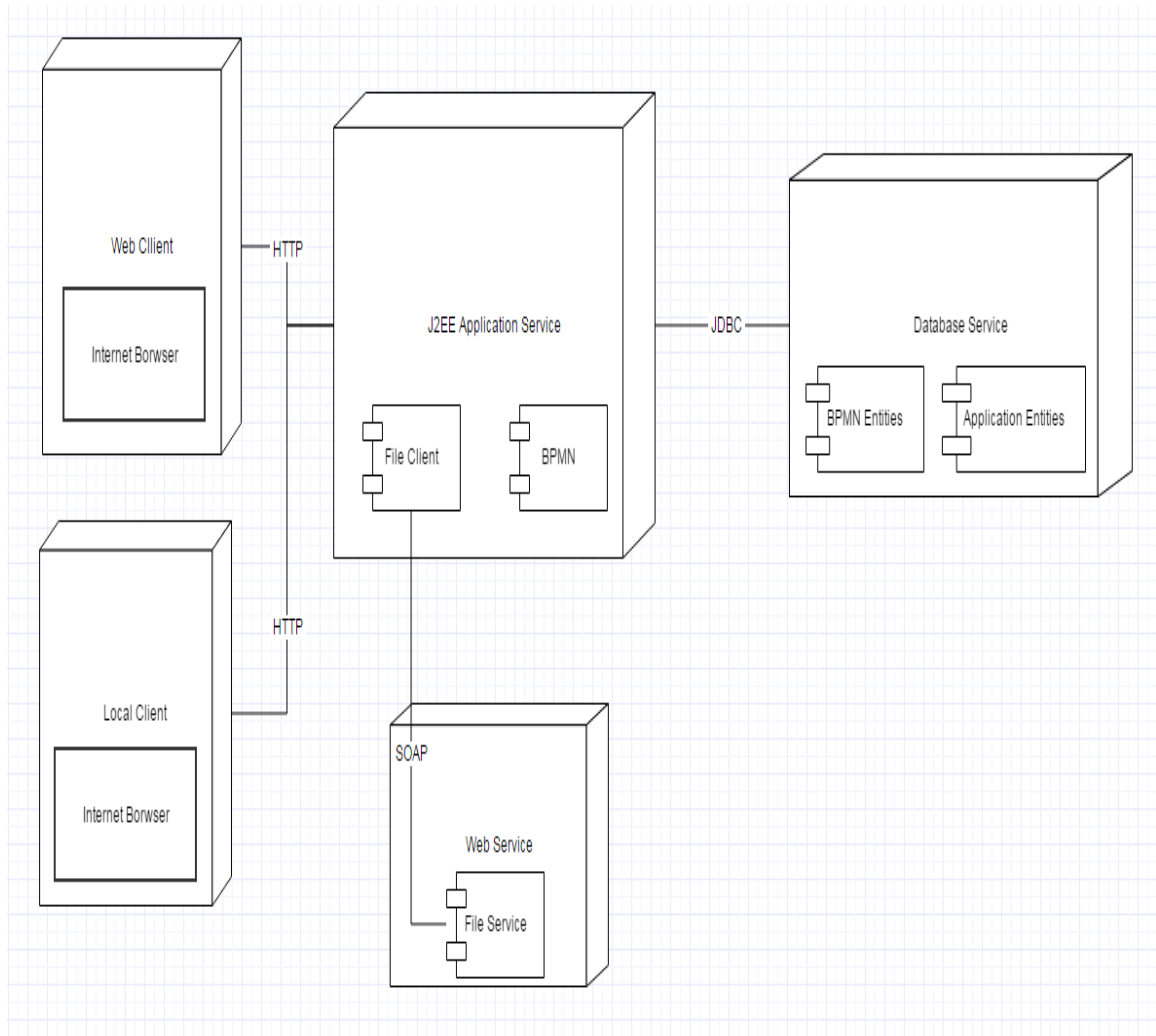


Figure 4.1. Overview of system construction.

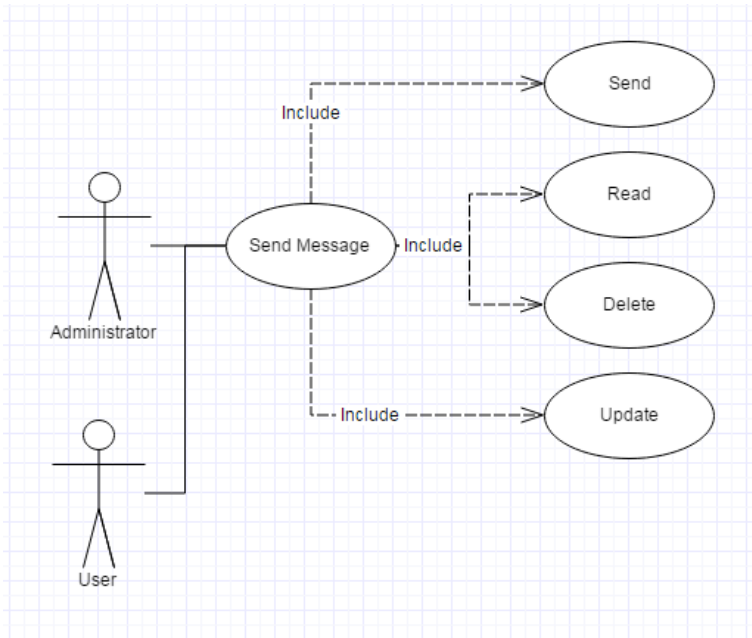


Figure 4.4. Message component use case.

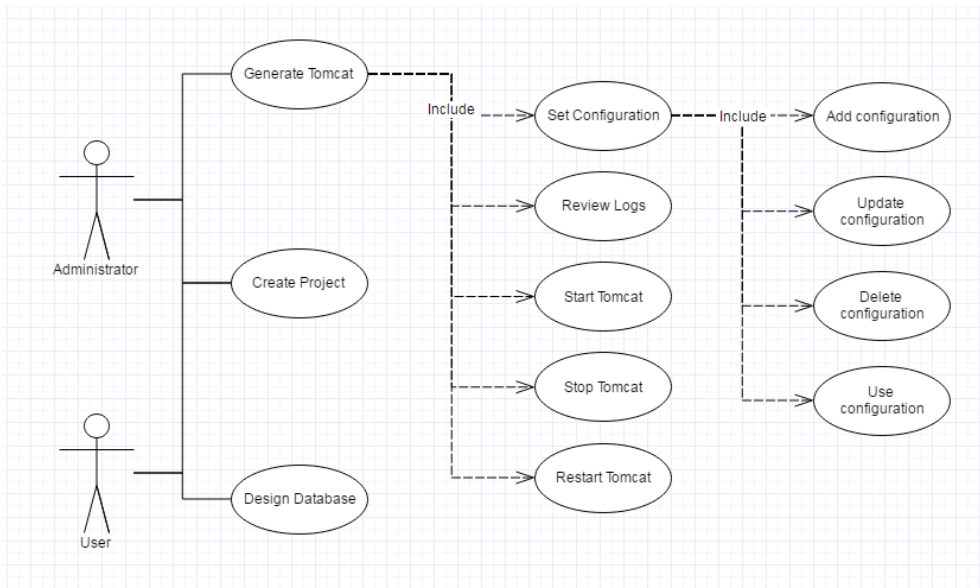


Figure 4.5. Tomcat component use case.

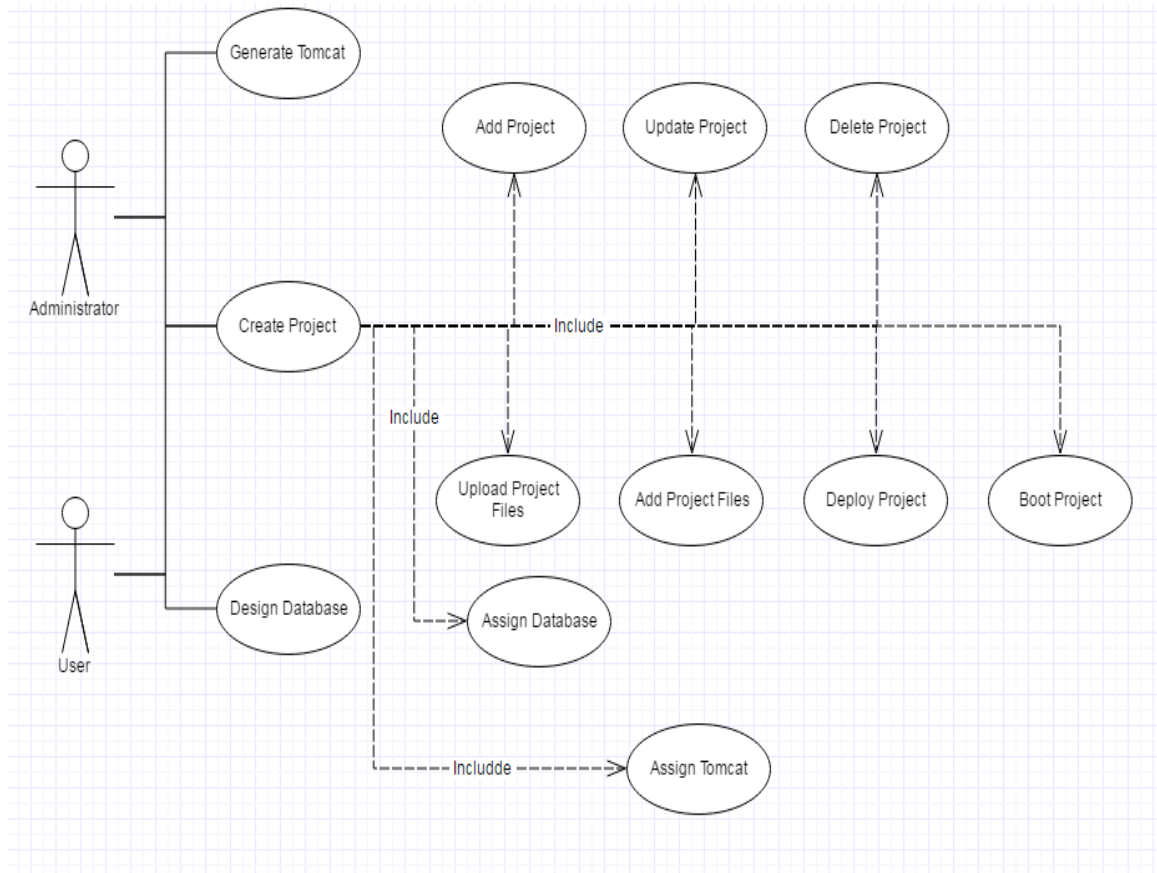


Figure 4.6. Project component use case.



Figure 4.7. Task and database component use case.

Main ER Diagrams

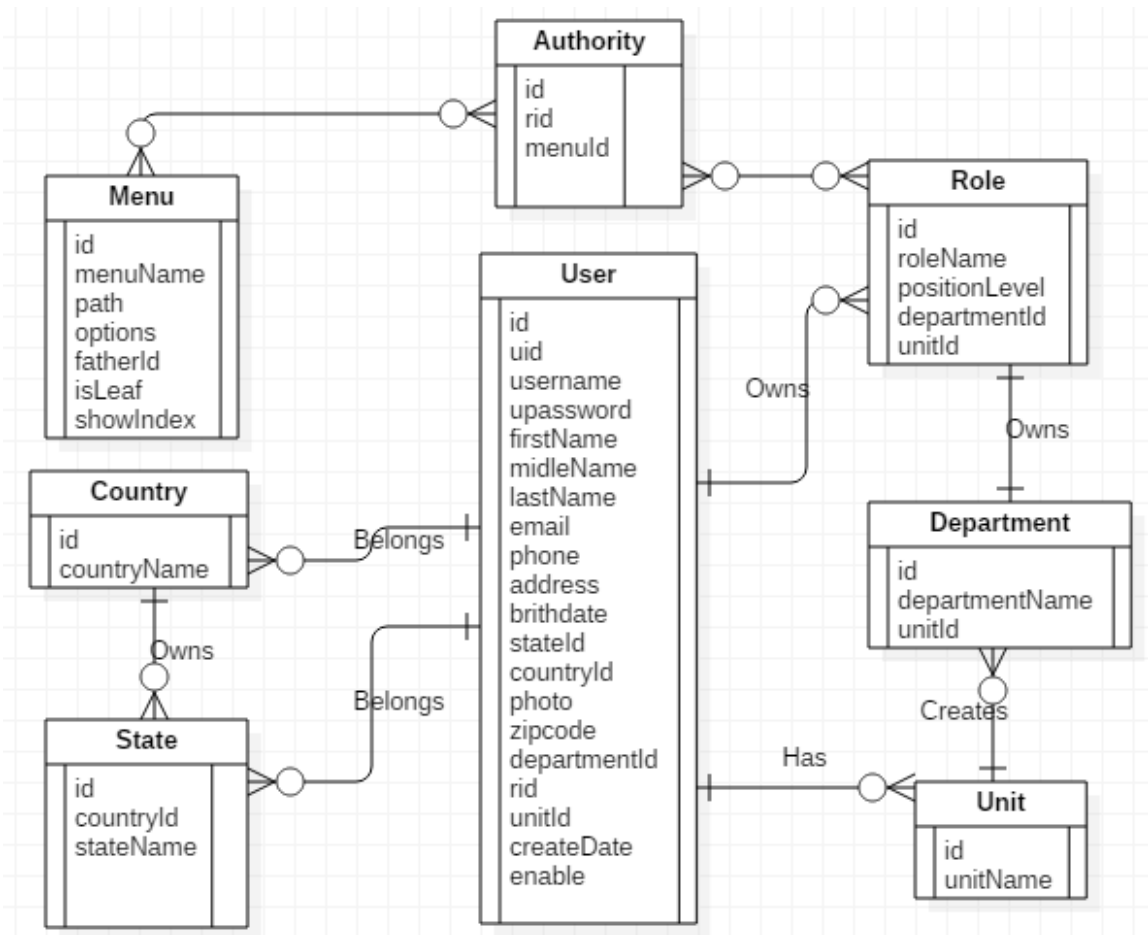


Figure 4.8. Customer ER component.

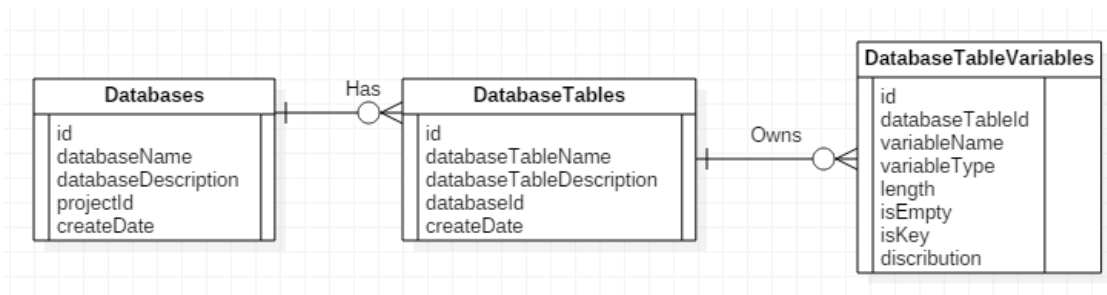


Figure 4.9. Database component.

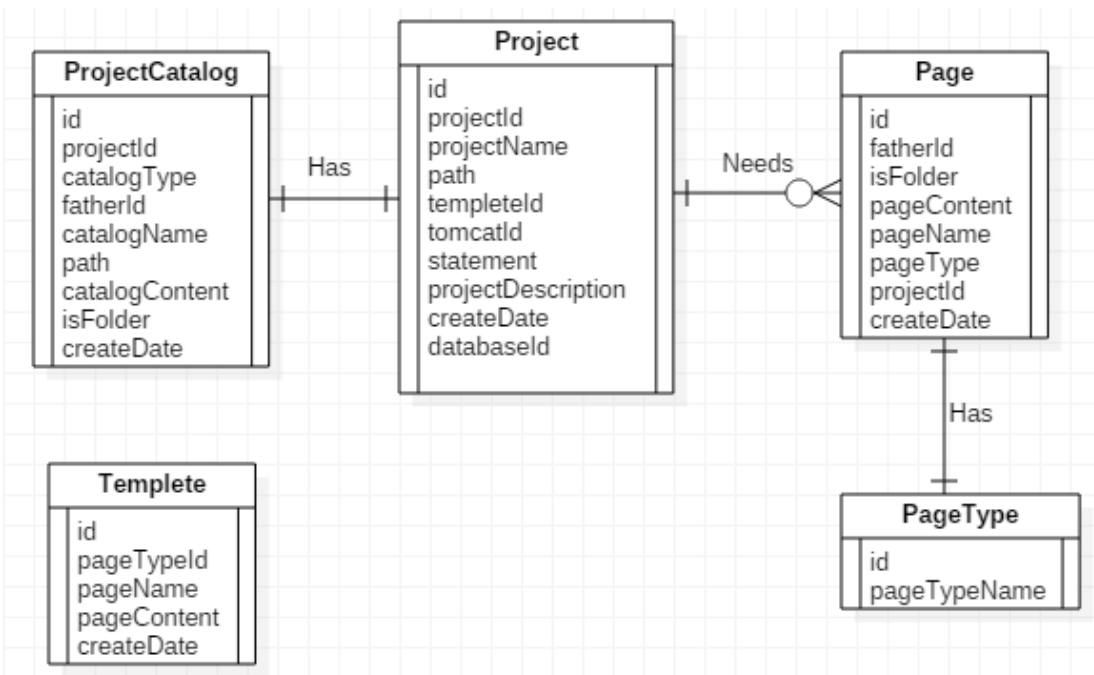


Figure 4.10. Project ER component.

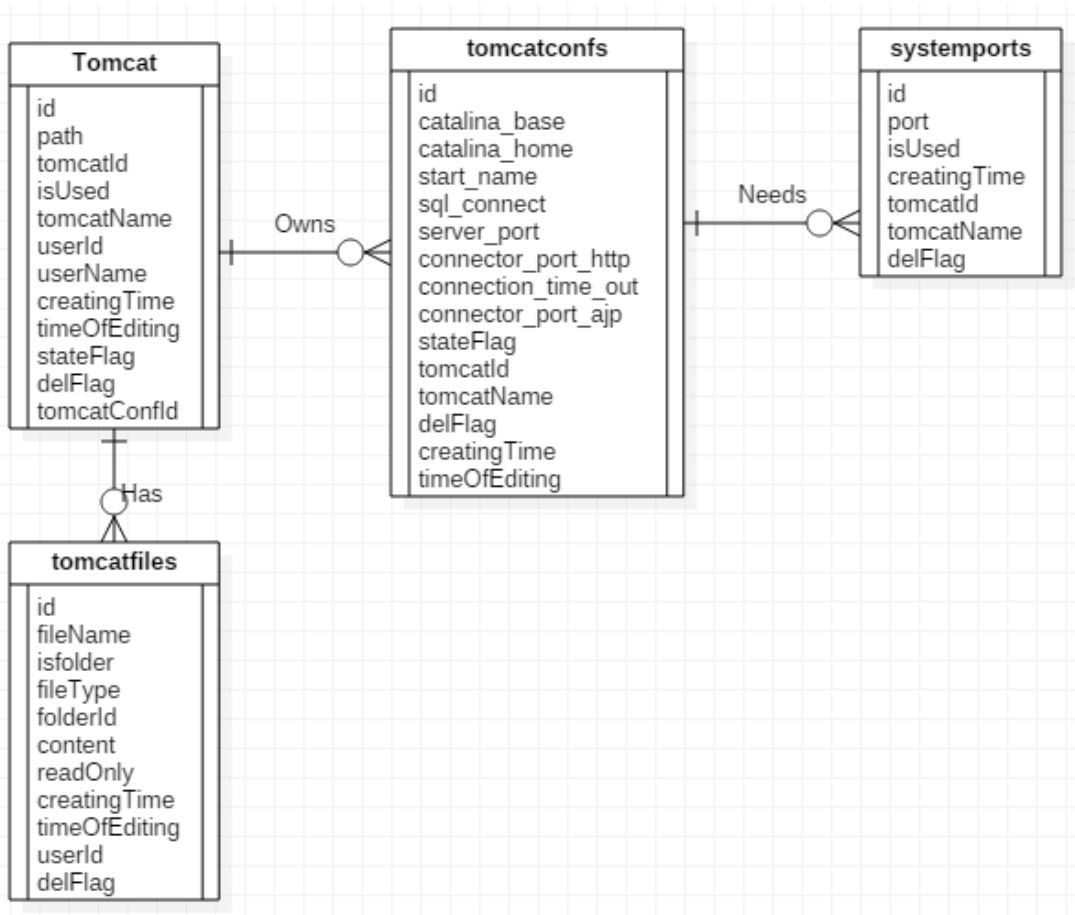


Figure 4.11. Tomcat ER component.

CHAPTER FIVE
PROJECT DATABASE


▶ id	int	11	0	<input type="checkbox"/>	
countryName	varchar	50	0	<input checked="" type="checkbox"/>	

Figure 5.1. Table of country.


▶ id	int	11	0	<input type="checkbox"/>	
countryId	int	11	0	<input checked="" type="checkbox"/>	
stateName	varchar	50	0	<input checked="" type="checkbox"/>	

Figure 5.2. Table of state.


▶ id	int	11	0	<input type="checkbox"/>	
departmentName	varchar	50	0	<input type="checkbox"/>	
unitId	int	11	0	<input type="checkbox"/>	

Figure 5.3. Table of department.


▶ id	int	11	0	<input type="checkbox"/>	
unitName	varchar	50	0	<input checked="" type="checkbox"/>	

Figure 5.4. Table of unit.


▶ id	int	11	0	<input type="checkbox"/>	
rid	int	11	0	<input type="checkbox"/>	
menuId	int	11	0	<input type="checkbox"/>	

Figure 5.5. Table of authorizes.


▶ id	int	11	0	<input type="checkbox"/>	
variableTypeName	varchar	50	0	<input type="checkbox"/>	
variableName	varchar	255	0	<input checked="" type="checkbox"/>	

Figure 5.6. Table of variable.


▶ id	int	11	0	<input type="checkbox"/>	
roleName	varchar	50	0	<input type="checkbox"/>	
positionLevel	int	11	0	<input type="checkbox"/>	
departmentId	int	11	0	<input type="checkbox"/>	
unitId	int	11	0	<input type="checkbox"/>	

Figure 5.7. Table of role.


▶ id	int	11	0	<input type="checkbox"/>	
menuName	varchar	50	0	<input type="checkbox"/>	
path	varchar	50	0	<input checked="" type="checkbox"/>	
options	varchar	255	0	<input checked="" type="checkbox"/>	
fatherId	int	11	0	<input type="checkbox"/>	
isLeaf	int	11	0	<input type="checkbox"/>	
showIndex	int	11	0	<input type="checkbox"/>	

Figure 5.8. Table of menu.

▶ id	int	11	0	<input type="checkbox"/>	
taskTypeName	varchar	50	0	<input type="checkbox"/>	

Figure 5.9. Table of task type.


▶ id	int	11	0	<input type="checkbox"/>	
pageTypeName	varchar	50	0	<input type="checkbox"/>	

Figure 4.10. Table of page type.


▶ id	int	11	0	<input type="checkbox"/>	
port	varchar	255	0	<input type="checkbox"/>	
isUsed	int	11	0	<input type="checkbox"/>	
creatingTime	varchar	255	0	<input type="checkbox"/>	
tomcatId	varchar	255	0	<input checked="" type="checkbox"/>	
tomcatName	varchar	255	0	<input checked="" type="checkbox"/>	
delFlag	int	11	0	<input type="checkbox"/>	

Figure 5.11. Table of system port.


▶ id	int	11	0	<input type="checkbox"/>	
originalTomcatPath	varchar	100	0	<input checked="" type="checkbox"/>	
targetTomcatPath	varchar	100	0	<input checked="" type="checkbox"/>	
projectPath	varchar	100	0	<input checked="" type="checkbox"/>	
photoPath	varchar	100	0	<input checked="" type="checkbox"/>	
bpmnProcessFilePath	varchar	100	0	<input checked="" type="checkbox"/>	
pageSize	int	11	0	<input checked="" type="checkbox"/>	

Figure 5.12. Table of system variable configuration.


▶ id	int	11	0	<input type="checkbox"/>	
uid	varchar	50	0	<input type="checkbox"/>	
username	varchar	50	0	<input type="checkbox"/>	
upassword	varchar	50	0	<input type="checkbox"/>	
firstName	varchar	50	0	<input checked="" type="checkbox"/>	
middleName	varchar	50	0	<input checked="" type="checkbox"/>	
lastName	varchar	50	0	<input checked="" type="checkbox"/>	
email	varchar	50	0	<input type="checkbox"/>	
phone	varchar	50	0	<input checked="" type="checkbox"/>	
address	varchar	100	0	<input checked="" type="checkbox"/>	
brithdate	varchar	50	0	<input checked="" type="checkbox"/>	
stateId	int	11	0	<input checked="" type="checkbox"/>	
countryId	int	11	0	<input checked="" type="checkbox"/>	
photo	varchar	100	0	<input checked="" type="checkbox"/>	
zipcode	varchar	50	0	<input checked="" type="checkbox"/>	
departmentId	int	50	0	<input checked="" type="checkbox"/>	
rid	int	11	0	<input type="checkbox"/>	
unitId	int	11	0	<input type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	
enable	int	11	0	<input type="checkbox"/>	

Figure 5.13. Table of customer.


▶ id	int	11	0	<input type="checkbox"/>	
pageTypeId	int	11	0	<input type="checkbox"/>	
pageName	varchar	50	0	<input type="checkbox"/>	
pageContent	text	0	0	<input checked="" type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	

Figure 5.14. Table of page type.


▶ id	int	11	0	<input type="checkbox"/>	
taskName	varchar	50	0	<input type="checkbox"/>	
receiverId	varchar	50	0	<input type="checkbox"/>	
departmentId	int	11	0	<input type="checkbox"/>	
taskFromId	varchar	50	0	<input type="checkbox"/>	
taskTypeId	int	11	0	<input type="checkbox"/>	
taskContent	text	0	0	<input type="checkbox"/>	
isAccept	int	11	0	<input checked="" type="checkbox"/>	
startDate	varchar	50	0	<input checked="" type="checkbox"/>	
dueDate	varchar	50	0	<input type="checkbox"/>	
process	int	11	0	<input checked="" type="checkbox"/>	
createDate	varchar	255	0	<input type="checkbox"/>	

Figure 5.15. Table of task.


▶ id	int	11	0	<input type="checkbox"/>	
projectId	varchar	50	0	<input checked="" type="checkbox"/>	
projectName	varchar	50	0	<input type="checkbox"/>	
path	varchar	255	0	<input type="checkbox"/>	
templeteId	int	11	0	<input checked="" type="checkbox"/>	
tomcatId	varchar	50	0	<input checked="" type="checkbox"/>	
statement	int	11	0	<input checked="" type="checkbox"/>	
projectDescription	text	0	0	<input checked="" type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	
databaseId	int	11	0	<input checked="" type="checkbox"/>	

Figure 5.16. Table of project.


▶ id	int	11	0	<input type="checkbox"/>	
projectId	int	11	0	<input type="checkbox"/>	
catalogType	varchar	50	0	<input type="checkbox"/>	
fatherId	int	11	0	<input type="checkbox"/>	
catalogName	varchar	50	0	<input type="checkbox"/>	
path	varchar	255	0	<input checked="" type="checkbox"/>	
catalogContent	longtext	0	0	<input checked="" type="checkbox"/>	
isFolder	int	11	0	<input type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	

Figure 5.17. Table of project file.


▶ id	int	11	0	<input type="checkbox"/>	
accountName	varchar	50	0	<input checked="" type="checkbox"/>	
address	varchar	100	0	<input checked="" type="checkbox"/>	
cpassword	varchar	50	0	<input checked="" type="checkbox"/>	
countryId	int	11	0	<input type="checkbox"/>	
createDate	varchar	50	0	<input checked="" type="checkbox"/>	
customerId	varchar	50	0	<input checked="" type="checkbox"/>	
DOB	varchar	50	0	<input checked="" type="checkbox"/>	
email	varchar	50	0	<input checked="" type="checkbox"/>	
enable	int	11	0	<input type="checkbox"/>	
firstName	varchar	50	0	<input checked="" type="checkbox"/>	
midleName	varchar	50	0	<input checked="" type="checkbox"/>	
lastName	varchar	50	0	<input checked="" type="checkbox"/>	
phone	varchar	50	0	<input checked="" type="checkbox"/>	
stateId	int	11	0	<input type="checkbox"/>	
zipcode	varchar	50	0	<input checked="" type="checkbox"/>	
apassword	varchar	255	0	<input checked="" type="checkbox"/>	
dateofbrith	varchar	255	0	<input checked="" type="checkbox"/>	

Figure 5.18. Table of customer.


▶ id	int	11	0	<input type="checkbox"/>	
bpmnProcessName	varchar	50	0	<input type="checkbox"/>	
realName	varchar	50	0	<input type="checkbox"/>	
bpmnProcessPath	varchar	100	0	<input type="checkbox"/>	
bpmnDescription	text	0	0	<input checked="" type="checkbox"/>	
uploadDate	varchar	50	0	<input type="checkbox"/>	

Figure 5.19. Table of upload BPMN file.


▶ id	int	11	0	<input type="checkbox"/>	
databaseName	varchar	50	0	<input type="checkbox"/>	
databaseDescription	text	0	0	<input type="checkbox"/>	
projectId	int	11	0	<input checked="" type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	

Figure 5.20. Table of database.


▶ id	int	11	0	<input type="checkbox"/>	
databaseTableName	varchar	255	0	<input type="checkbox"/>	
databaseTableDescription	text	0	0	<input checked="" type="checkbox"/>	
databaseId	int	11	0	<input type="checkbox"/>	
createDate	varchar	255	0	<input type="checkbox"/>	

Figure 5.21. Table of database table.


▶ id	int	11	0	<input type="checkbox"/>	
databaseTableId	int	11	0	<input type="checkbox"/>	
variableName	varchar	50	0	<input type="checkbox"/>	
variableType	varchar	50	0	<input type="checkbox"/>	
length	int	11	0	<input type="checkbox"/>	
isEmpty	int	11	0	<input type="checkbox"/>	
isKey	int	11	0	<input type="checkbox"/>	
discription	text	0	0	<input checked="" type="checkbox"/>	

Figure 5.22. Table of database table variable.


▶ id	int	11	0	<input type="checkbox"/>	
createDate	varchar	255	0	<input checked="" type="checkbox"/>	
fatherId	int	11	0	<input type="checkbox"/>	
isFolder	int	11	0	<input type="checkbox"/>	
pageContent	varchar	255	0	<input checked="" type="checkbox"/>	
pageName	varchar	255	0	<input checked="" type="checkbox"/>	
pageType	varchar	255	0	<input checked="" type="checkbox"/>	
projectId	int	11	0	<input type="checkbox"/>	

Figure 5.23. Table of page.


▶ id	int	11	0	<input type="checkbox"/>	
title	varchar	50	0	<input type="checkbox"/>	
senderName	varchar	50	0	<input checked="" type="checkbox"/>	
senderId	varchar	50	0	<input checked="" type="checkbox"/>	
receiverName	varchar	50	0	<input checked="" type="checkbox"/>	
receiverId	varchar	50	0	<input checked="" type="checkbox"/>	
content	text	0	0	<input checked="" type="checkbox"/>	
isRead	int	11	0	<input type="checkbox"/>	
isReply	int	50	0	<input type="checkbox"/>	
replyId	int	50	0	<input type="checkbox"/>	
createDate	varchar	50	0	<input type="checkbox"/>	

Figure 5.24. Table of message.


▶ id	int	11	0	<input type="checkbox"/>	
catalina_base	varchar	255	0	<input type="checkbox"/>	
catalina_home	varchar	255	0	<input type="checkbox"/>	
start_name	varchar	255	0	<input type="checkbox"/>	
sql_connect	text	0	0	<input checked="" type="checkbox"/>	
server_port	varchar	255	0	<input type="checkbox"/>	
connector_port_http	varchar	255	0	<input type="checkbox"/>	
connection_time_out	varchar	255	0	<input type="checkbox"/>	
connector_port_ajp	varchar	255	0	<input type="checkbox"/>	
stateFlag	int	11	0	<input type="checkbox"/>	
tomcatId	varchar	255	0	<input type="checkbox"/>	
tomcatName	varchar	255	0	<input type="checkbox"/>	
delFlag	int	11	0	<input type="checkbox"/>	
creatingTime	varchar	255	0	<input type="checkbox"/>	
timeOfEditing	varchar	255	0	<input type="checkbox"/>	

Figure 5.25. Table of tomcat config.


▶ id	varchar	255	0	<input type="checkbox"/>	
fileName	varchar	50	0	<input type="checkbox"/>	
isfolder	int	11	0	<input type="checkbox"/>	
fileType	int	11	0	<input checked="" type="checkbox"/>	
folderId	varchar	255	0	<input type="checkbox"/>	
content	text	0	0	<input checked="" type="checkbox"/>	
readOnly	int	11	0	<input type="checkbox"/>	
creatingTime	varchar	50	0	<input type="checkbox"/>	
timeOfEditing	varchar	50	0	<input type="checkbox"/>	
userId	int	11	0	<input type="checkbox"/>	
delFlag	int	11	0	<input type="checkbox"/>	

Figure 5.26. Table of file.


▶ id	int	11	0	<input type="checkbox"/>	
path	varchar	255	0	<input type="checkbox"/>	
tomcatId	varchar	255	0	<input type="checkbox"/>	
isUsed	int	11	0	<input type="checkbox"/>	
tomcatName	varchar	255	0	<input type="checkbox"/>	
userId	int	11	0	<input type="checkbox"/>	
userName	varchar	50	0	<input type="checkbox"/>	
creatingTime	varchar	50	0	<input type="checkbox"/>	
timeOfEditing	varchar	50	0	<input type="checkbox"/>	
stateFlag	int	11	0	<input type="checkbox"/>	
delFlag	int	11	0	<input type="checkbox"/>	
tomcatConfId	int	11	0	<input checked="" type="checkbox"/>	

Figure 5.27. Table of tomcat.

CHAPTER SIX

PROJECT COMPONENT

System Configuration

When login into the system, the system administrator needs to setup several items as Figure 6.1 shows. The first item is the root folder path of original tomcat and target tomcat folder path where customer tomcat should be saved at. The original tomcat folder has been put with the project together deployed to tomcat service project folder. The second item is to setup the project path where customer's project should be stored into. The third item is to setup photo path where all upload pictures should be saved. The fourth item is BPMN file path where all upload BPMN process files go to and where the BPMN engine should look for deployed process files. The last one is page size that can determine how many items should show on a list page.

Original Tomcat Path *	D:/Apache Tomcat/webapps/WebFileServer/WEB-INF/TomcatSource/Apache
Target Tomcat Path *	D:/Apache Tomcat/webapps/WebFileServer/WEB-INF/TomcatSource/
Project Path *	resources/assets/projects/
Photo Path *	resources/assets/uploadFiles/
Bpmn Process File Path *	resources/assets/bpmnFile/
Page Size *	10

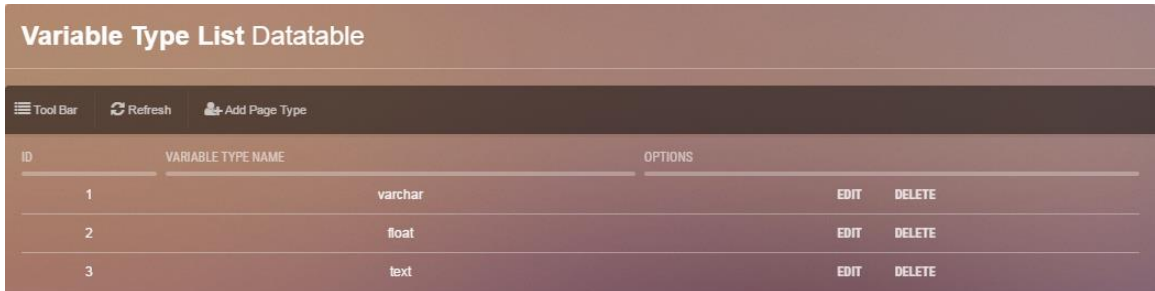
Figure 6.1. System variable configuration.

The system port configuration (Figure 6.2) is for tomcat web service. Every tomcat web service needs three system ports so that the system administrator needs to check and make sure that there has enough system ports for the customer to use. Otherwise, the customers cannot create tomcat web service.

System Port List Datable						
ID	IS USED	PORT NAME	TOMMCAT ID	TOMCAT NAME	OPTIONS	
1	Yes	9001	afc2ed23-87f9-488b-9657-f1b0c642c570	TomcatServer11	EDIT	DELETE
2	No	9002	None	None	EDIT	DELETE
3	Yes	9003	afc2ed23-87f9-488b-9657-f1b0c642c570	TomcatServer11	EDIT	DELETE
4	Yes	9004	afc2ed23-87f9-488b-9657-f1b0c642c570	TomcatServer11	EDIT	DELETE
5	No	9005	None	None	EDIT	DELETE
6	No	9006	None	None	EDIT	DELETE
7	No	9010	None	None	EDIT	DELETE
8	No	9011	None	None	EDIT	DELETE
9	No	9012	None	None	EDIT	DELETE
10	No	9020	None	None	EDIT	DELETE

Figure 6.2. System port configuration.

The variable type (Figure 6.3) is for creating databases. Customers can create databases through this project and they need these variables type to declare every element.



ID	VARIABLE TYPE NAME	OPTIONS
1	varchar	EDIT DELETE
2	float	EDIT DELETE
3	text	EDIT DELETE

Figure 6.3. Variable type configuration.

The page type configuration (Figure 6.4) is for creating web pages. Customers can create project file trees with different web file types and folders such as JSP, js, or CSS. If the file is a JSP, then customers can write those files in JSP language. After customers deploy their projects, all file will be deployed to their tomcat web services as well.

Page Type List				
ID	PAGE TYPE NAME	OPTIONS		
1	Home Page	EDIT	DELETE	
2	Folder	EDIT	DELETE	
3	File	EDIT	DELETE	
4	Jsp	EDIT	DELETE	
5	HTML	EDIT	DELETE	
6	EXTJS	EDIT	DELETE	

Figure 6.4. Page type configuration.

The task type configuration (Figure 6.5) is for BPMN task. The customers can assign different type of BPMN tasks to other customers.

Task Type List				
ID	TASK TYPE NAME	OPTIONS		
3	Daily Task	EDIT	DELETE	
4	Weekly Task	EDIT	DELETE	
5	Urgen	EDIT	DELETE	

Figure 6.5. Task type configuration.

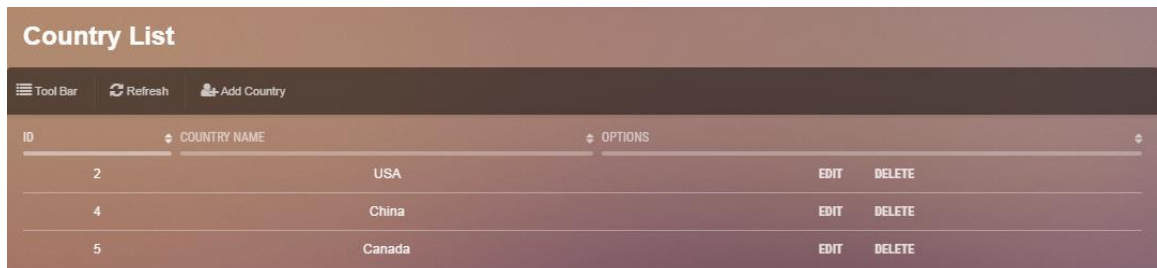
Setup Customer Model

To set up a company account, the system administrator needs to use country and state configuration (Figure 6.6 and 3.7).



ID	COUNTRY NAME	STATE NAME	OPTIONS
1	USA	CA	EDIT DELETE
2	USA	NY	EDIT DELETE
3	China	Shanghai	EDIT DELETE
4	Canada	JJJ	EDIT DELETE

Figure 6.6. State configuration.



ID	COUNTRY NAME	OPTIONS
2	USA	EDIT DELETE
4	China	EDIT DELETE
5	Canada	EDIT DELETE

Figure 6.7. Country configuration.

The company configuration (Figure 6.8) is used to separate different companies. When enterprise customers sign in the system, they will not see each other.

Unit List				
ID	UNIT NAME	OPTIONS		
1	Company X	EDIT	DELETE	
2	Company Y	EDIT	DELETE	
3	Company Z	EDIT	DELETE	

Figure 6.8. Company configuration.

As we know, every company has many departments. Every department has its own roles. Under this system, The number of menus customers can access are controlled by the role configuration. Every company has a system level administrator who can access the role (Figure 6.9), department (Figure 6.10), and menu (Figure 6.11) tab, but the administrator can only allocate the rights he/she has.

Role List				
ID	ROLE NAME	POSITION LEVEL	DEPARTMENT NAME	OPTIONS
2	SystemAdmin	Level 1	IT	EDIT DELETE
16	DepartmentManager	Level 2	IT	EDIT DELETE
17	HRManager	Level 1	HR	EDIT DELETE

Figure 6.9. Role configuration.

Department Information							
☰ Tool Bar 🔄 Refresh ➕ Add Department							
ID	DEPARTMENT NAME	COMPANY NAME	OPTIONS				
1	IT	Company X	EDIT	DELETE			
7	MainOffice	Company X	EDIT	DELETE			
8	HR	Company X	EDIT	DELETE			

Figure 6.10. Department configuration.

Menu Information							
☰ Tool Bar 🔄 Refresh ➕ Add Menu							
ID	MENU NAME	PATH	FATHER	ISCHILD	SHOW INDEX	OPTIONS	
1	AdminManagement	None	None	NO	1	EDIT	DELETE
3	AdminList	/admin/adminList	AdminManagement	Yes	1	EDIT	DELETE
4	RoleList	/admin/roleList	AdminManagement	Yes	2	EDIT	DELETE
5	DepartmentList	/admin/departmentList	AdminManagement	Yes	3	EDIT	DELETE
6	CountryList	/admin/countryList	AdminManagement	Yes	4	EDIT	DELETE
7	StateList	/admin/stateList	AdminManagement	Yes	5	EDIT	DELETE
8	TaskTypeList	/admin/taskTypeList	AdminManagement	Yes	6	EDIT	DELETE
11	MenuList	/admin/menuList	AdminManagement	Yes	7	EDIT	DELETE

Figure 6.11. Country configuration.

Setup Tomcat Web Service

To add tomcat services, the customers need to go to the project management tab. Under that tab, the customers can find tomcat web service list (Figure 6.12). To add tomcats, the customers need to click add tomcat button at the toolbar of a display list, then the customers will see add tomcat page (Figure 6.). Now, they need to input a unique tomcat web service name, because the system will check if the name is a unique name.

ID	TOMCAT ID	TOMCAT NAME	USER NAME	USED	STATEMENT	OPTIONS
10	atc2ed23-87f9-488b-9657-41b0c642c570	TomcatServer11	admin	Yes	Stop	CONF FILE START STOP RE-START LOG EDIT DELETE

Figure 6.12. Tomcat web service list.

Figure 6.13. Add tomcat web service page.

When the customers add their tomcat web service successfully, the customers are able to get into their tomcat fie configuration page (Figure 6.14). The customers then will see the tomcat configuration list. On the list, the customers can add many configuration files and can only choose one of them to apply to the tomcat. After then, customers will see very basic information of the tomcat configuration (Figure 6.15), including server port, HTTP port, ALP port, and current tomcat configuration status.

ID	TOMCAT NAME	SERVER PORT	HTTP PORT	TIME OUT PORT	AJP PORT	STATEMENT	OPTIONS
6	TomcatServer11	9003	9004	333	9001	Used	USE EDIT DELETE

Figure 6.14. Tomcat configuration list.

Tomcat ID	afc2ed23-87f9-488b-9657-f1b0c642c570
Catalina Base	D:/Apache Tomcat/webapps/WebFileServer/WEB-INF/TomcatSource/afc2ed23-87f9-488b-9657-f1b0c642c570
Catalina Home	D:/Apache Tomcat/webapps/WebFileServer/WEB-INF/TomcatSource/afc2ed23-87f9-488b-9657-f1b0c642c570
Sql Connect	
Server Port *	Please choose a server port ▼
Connector Http Port *	Please choose a http port ▼
Connector Ajp Port *	Please choose a ajp port ▼
Connection Time Out *	Connection Time Out

[Submit](#) [Reset](#) [Back to Tomcat Configuration List](#)

Figure 6.15. Add tomcat web service page.

After the customer tomcat servers are created, the customers can see the tomcat file tree at tomcat file list page (Figure 6.16). At there, the customers can also see a physical file as Figure 6.17 shows.

Tomcat File List

[Back](#)
[Refresh](#)
[Add Tomcat Folder](#)
[Add Tomcat File](#)

ID	FILE NAME	TYPE	SYSTEM FILES	CREATED TIME	OPTIONS
11f80975-c263-427e-a38b-6d32b7b3370b	webapps		Yes	2016-10-23 15:34:31	ACCESS
94587246-3269-4afc-ab19-b69ee4594a9d	logs		Yes	2016-10-23 15:34:31	ACCESS
bae778cf-f57c-441c-b8af-993a01f6ed82	conf		Yes	2016-10-23 15:34:31	ACCESS
f1bac391-a3c9-47f1-a82e-bcfa60dd8e07	bin		Yes	2016-10-23 15:34:31	ACCESS

Figure 6.16. Tomcat configuration list.

catalinaproperties Content

2016-10-23 15:34:31

[Back](#)

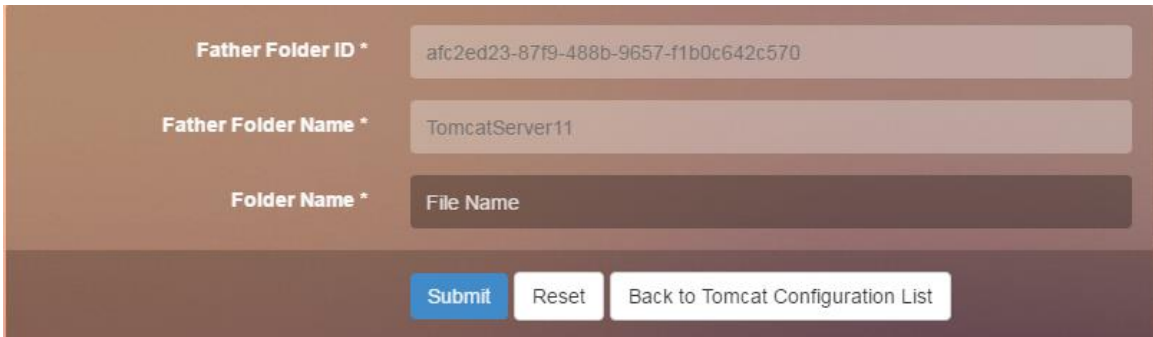
```

# Licensed to the Apache Software Foundation (ASF) under one or more # contributor license agreements. See the NOTICE file distributed with # this work for
additional information regarding copyright ownership. # The ASF licenses this file to You under the Apache License, Version 2.0 # (the "License"); you may not use
this file except in compliance with # the License. You may obtain a copy of the License at # # http://www.apache.org/licenses/LICENSE-2.0 # # Unless required by
applicable law or agreed to in writing, software # distributed under the License is distributed on an "AS IS" BASIS, # WITHOUT WARRANTIES OR CONDITIONS
OF ANY KIND, either express or implied. # See the License for the specific language governing permissions and # limitations under the License. # # List of
comma-separated packages that start with or equal this string # will cause a security exception to be thrown when # passed to checkPackageAccess unless the #
corresponding RuntimePermission ("accessClassInPackage."+package) has # been granted.
package.access=sun.,org.apache.catalina.,org.apache.coyote.,org.apache.tomcat.,org.apache.jasper.,sun.beans. # # List of comma-separated packages that
start with or equal this string # will cause a security exception to be thrown when # passed to checkPackageDefinition unless the # corresponding
RuntimePermission ("defineClassInPackage."+package) has # been granted. # # by default, no packages are restricted for definition, and none of # the class
loaders supplied with the JDK call checkPackageDefinition. # package.definition=sun.,java.,org.apache.catalina.,org.apache.coyote.,
package.definition=sun.,java.,org.apache.catalina.,org.apache.coyote.,org.apache.tomcat.,org.apache.jasper. # # # List of comma-separated paths defining the
contents of the "common" # classloader. Prefixes should be used to define what is the repository type. # Path may be relative to the CATALINA_HOME or
CATALINA_BASE path or absolute. # If left as blank, the JVM system loader will be used as Catalinas "common" # loader. # Examples: # "foo": Add this folder as a
class repository # "foo/*.jar": Add all the JARs of the specified folder as class # repositories # "foo/bar.jar": Add bar.jar as a class repository
common.loader=${catalina.home}/lib,${catalina.home}/lib/*.jar # # List of comma-separated paths defining the contents of the "server" # classloader. Prefixes
should be used to define what is the repository type. # Path may be relative to the CATALINA_HOME or CATALINA_BASE path or absolute. # If left as blank, the
"common" loader will be used as Catalinas "server" # loader. # Examples: # "foo": Add this folder as a class repository # "foo/*.jar": Add all the JARs of the
specified folder as class # repositories # "foo/bar.jar": Add bar.jar as a class repository server.loader= # # List of comma-separated paths defining the contents of
the "shared" # classloader. Prefixes should be used to define what is the repository type. # Path may be relative to the CATALINA_BASE path or absolute. If left as
blank, # the "common" loader will be used as Catalinas "shared" loader. # Examples: # "foo": Add this folder as a class repository # "foo/*.jar": Add all the JARs of
the specified folder as class # repositories # "foo/bar.jar": Add bar.jar as a class repository # Please note that for single jars, e.g. bar.jar, you need the URL form #
starting with file:. shared.loader= # # String cache configuration. tomcat.util.buf.StringCache.byte.enabled=true #tomcat.util.buf.StringCache.char.enabled=true
#tomcat.util.buf.StringCache.trainThreshold=500000 #tomcat.util.buf.StringCache.cacheSize=5000

```

Figure 6.17. Tomcat file content.

The customers can customize the tomcat file and folder as Figure 6.18 and 3.19 shows. The customers can also add an extra tomcat running jar into tomcat lib for their projects.



Father Folder ID *

Father Folder Name *

Folder Name *

Figure 6.18. Add tomcat folder.



Father Folder ID *

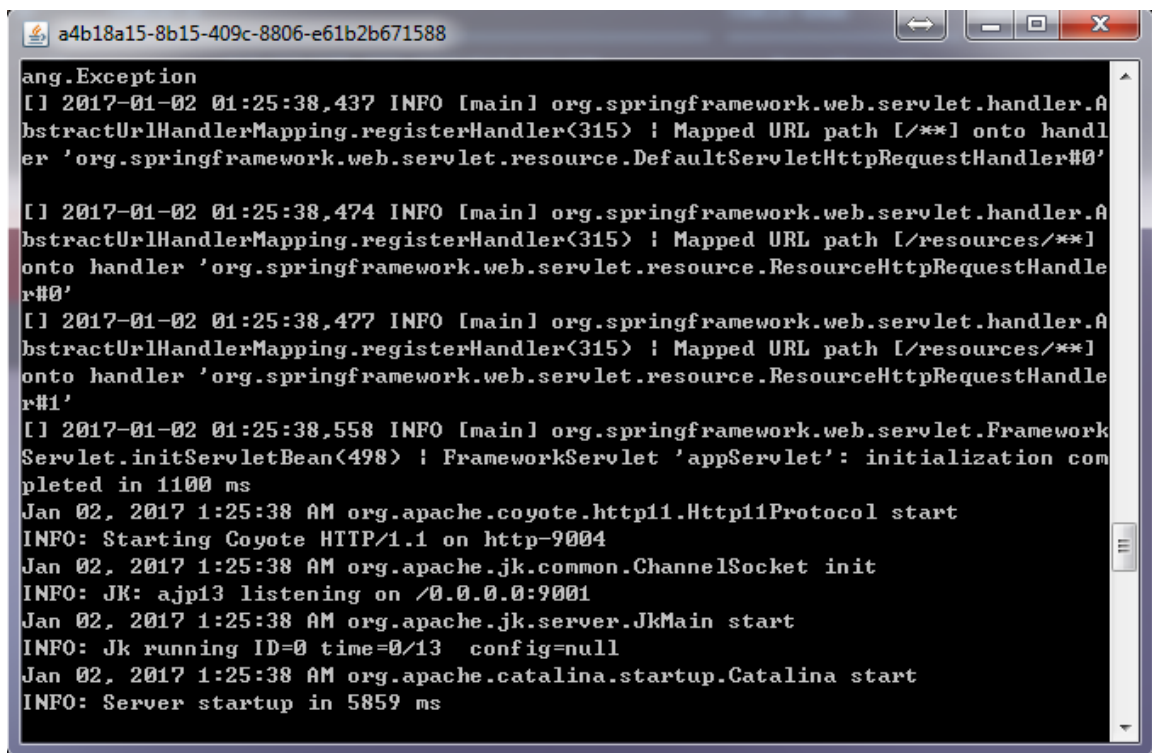
Father Folder Name *

File Name *

File Content *

Figure 6.19. Add tomcat file.

When everything has been setup, the customer can do these tomcat operations that are start, stop, and re-start at the tomcat service list page.



```
ang.Exception
[] 2017-01-02 01:25:38,437 INFO [main] org.springframework.web.servlet.handler.A
bstractUrlHandlerMapping.registerHandler(315) ! Mapped URL path [/]**] onto handl
er 'org.springframework.web.servlet.resource.DefaultServletHttpRequestHandler#0'

[] 2017-01-02 01:25:38,474 INFO [main] org.springframework.web.servlet.handler.A
bstractUrlHandlerMapping.registerHandler(315) ! Mapped URL path [/resources/**]
onto handler 'org.springframework.web.servlet.resource.ResourceHttpRequestHandle
r#0'

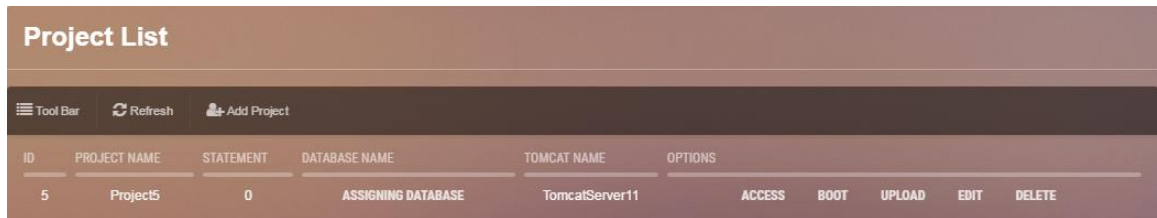
[] 2017-01-02 01:25:38,477 INFO [main] org.springframework.web.servlet.handler.A
bstractUrlHandlerMapping.registerHandler(315) ! Mapped URL path [/resources/**]
onto handler 'org.springframework.web.servlet.resource.ResourceHttpRequestHandle
r#1'

[] 2017-01-02 01:25:38,558 INFO [main] org.springframework.web.servlet.Framework
Servlet.initServletBean(498) ! FrameworkServlet 'appServlet': initialization com
pleted in 1100 ms
Jan 02, 2017 1:25:38 AM org.apache.coyote.http11.Http11Protocol start
INFO: Starting Coyote HTTP/1.1 on http-9004
Jan 02, 2017 1:25:38 AM org.apache.jk.common.ChannelSocket init
INFO: JK: ajp13 listening on /0.0.0.0:9001
Jan 02, 2017 1:25:38 AM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/13 config=null
Jan 02, 2017 1:25:38 AM org.apache.catalina.startup.Catalina start
INFO: Server startup in 5859 ms
```

Figure 6.19. Tomcat web service console.

Creating Project

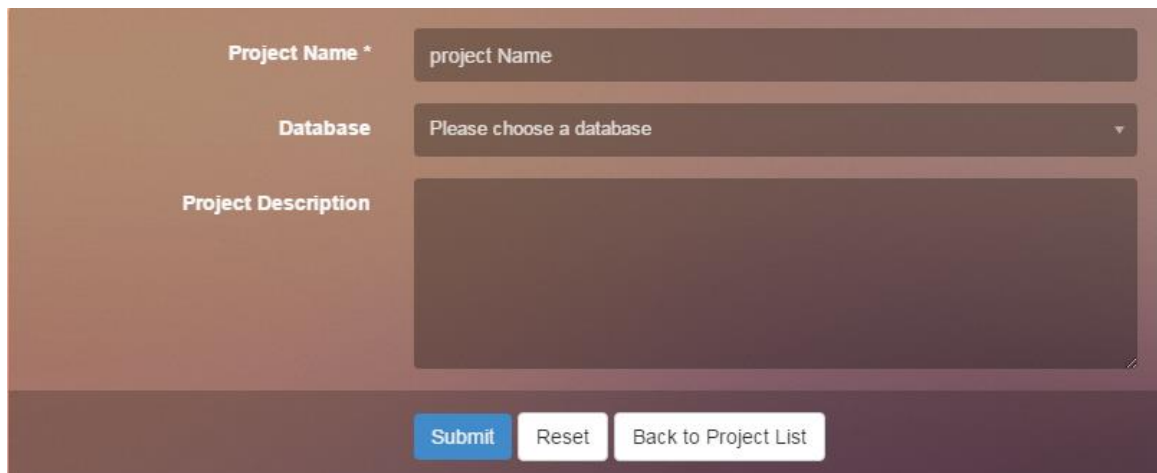
Under the project management, the customers can add projects to the project list (Figure 20) after the customers create tomcat web services.



ID	PROJECT NAME	STATEMENT	DATABASE NAME	TOMCAT NAME	OPTIONS
5	Project5	0	ASSIGNING DATABASE	TomcatServer11	ACCESS BOOT UPLOAD EDIT DELETE

Figure 6.20. Project list.

To add a project, customers need to input at least the project name and project description what the customer will write some note to explain what the project will do and what it is.



Project Name *

Database

Project Description

Figure 6.21. Add project page.

When the project has been created, the customer can determine what type of the project file he/she wants to do. If the customer chooses to create the project file with clicking add catalog button (Figure 22), the customer will create an online coding file. If the customer chooses to click the upload catalog button (Figure 22), the customer will upload a tomcat project war file to the tomcat web service and run that project.

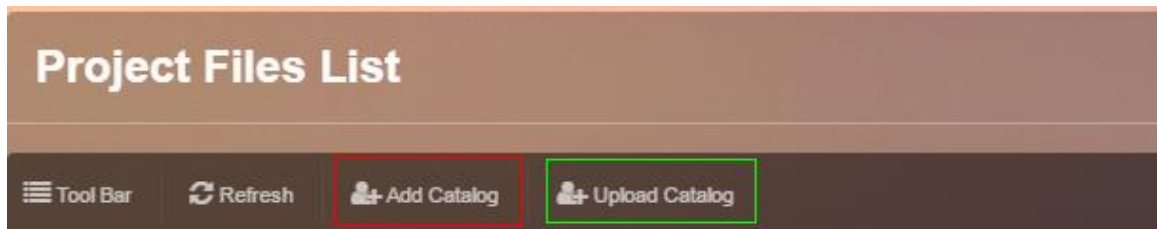


Figure 6.22 Two ways to add project files.

Deploying Project and Start Web Service

When the customer has his/her project file done, the customer needs to boot the project first. Boot the project (Figure 6.24) means project files the customer created online will be written to disk from database and then the customer needs to click upload button (Figure 6.24) to upload these files to tomcat web service. Then, the customer can start his/her tomcat web service successfully.

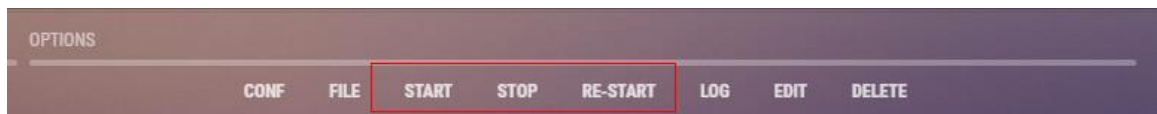


Figure 6.23. Tomcat web service options.

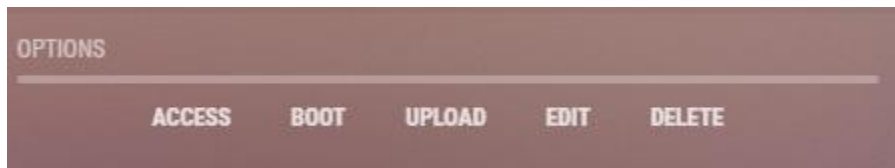
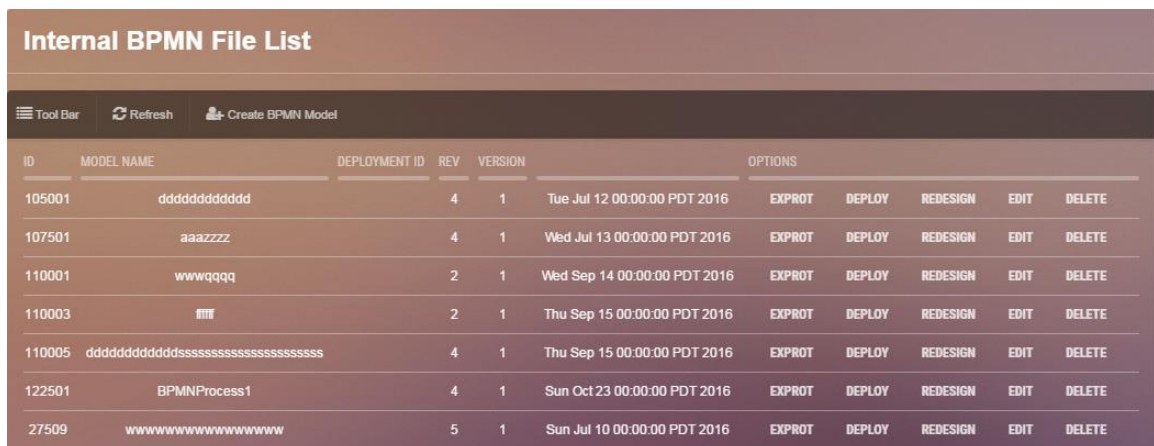


Figure 6.24. Project options.

Deploying BPMN Process Flow

To deploy a BPMN process file, a customer needs to determine which way he/she wants to use. I provide two ways to deploy BPMN process file. Under the project management tab, there have two BPMN display lists. The first list (Figure 6.25) will show a customer internal BPMN file list what all BPMN process files are created by the system with embedded BPMN file editor (Figure 6.30). Another list (Figure 6.26) is for the customer to upload and deploy a BPMN process file to BPMN engine. The external BPMN will be shown at external BPMN display list (Figure 6.27).



ID	MODEL NAME	DEPLOYMENT ID	REV	VERSION		OPTIONS
105001	ddddddddddddd		4	1	Tue Jul 12 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
107501	aaazzzz		4	1	Wed Jul 13 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
110001	wwwqqqq		2	1	Wed Sep 14 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
110003	tttt		2	1	Thu Sep 15 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
110005	dddddddddddddssssssssssssssssssss		4	1	Thu Sep 15 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
122501	BPMNProcess1		4	1	Sun Oct 23 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE
27509	wwwwwwwwwwwwwww		5	1	Sun Jul 10 00:00:00 PDT 2016	EXPROT DEPLOY REDESIGN EDIT DELETE

Figure 6.25. Created on system side BPMN process list.

Uploaded BPMN File List				
ID	BPMN PROCESS NAME	BPMN PROCESS PATH	OPTIONS	
11	qqq	resources/assets/bpmnFile/56022a7f-e663-46b8-8a3e-3a8f43a9849b.bpmn	DEPLOY	DELETE
12	www	resources/assets/bpmnFile/4b39cbdb-c3be-461e-a003-6ad6754e7d8b.bpmn	DEPLOY	DELETE
13	www	resources/assets/bpmnFile/56db4d25-decd-4461-bf57-08c0fb6d74d.bpmn	DEPLOY	DELETE
14	ddd	resources/assets/bpmnFile/72515930-1cab-4050-8c8b-39bc491840c.bpmn	DEPLOY	DELETE
15	2233	resources/assets/bpmnFile/eec3bfff-d2d3-49cf-902f-5489fae5b436.bpmn	DEPLOY	DELETE
16	eee	resources/assets/bpmnFile/02cccfe9-d117-48c9-b06b-133b362386c3.bpmn	DEPLOY	DELETE
17	ddd	resources/assets/bpmnFile/d56fee6b-190-4dba-b907-90d95967d39a.bpmn	DEPLOY	DELETE
18	fgggg	resources/assets/bpmnFile/6f91b165-56d3-4131-8310-1323ef48a7e2.bpmn	DEPLOY	DELETE
19	ttttt	resources/assets/bpmnFile/204035d0-2edd-4e1f-bf79-e6eaa45aedc.bpmn	DEPLOY	DELETE
20	777	resources/assets/bpmnFile/ae142dbb-c33f-4501-86af-cacffc4872da.bpmn	DEPLOY	DELETE

Figure 6.26. Uploaded BPMN process file list.

External BPMN File List								
PROCESSDEFINITIONID	DEPLOYMENTID	NAME	KEY	VERSION	XML	PIC	DEPLOYED DATE	OPTION
leave:10:65004	65001	Leave Process	leave	10	XML	PIC	Sun Jul 10 23:15:56 PDT 2016	EXPROT
leave:9:62504	62501	Leave Process	leave	9	XML	PIC	Sun Jul 10 23:07:01 PDT 2016	EXPROT
leave:8:52531	52528	Leave Process	leave	8	XML	PIC	Sun Jul 10 22:49:16 PDT 2016	EXPROT
leave:7:52509	52506	Leave Process	leave	7	XML	PIC	Sun Jul 10 22:45:30 PDT 2016	EXPROT
leave:6:25007	25004	Leave Process	leave	6	XML	PIC	Sun Jul 10 15:01:45 PDT 2016	EXPROT
leave:5:22504	22501	Leave Process	leave	5	XML	PIC	Sun Jul 10 14:43:04 PDT 2016	EXPROT
leave:4:17504	17501	Leave Process	leave	4	XML	PIC	Sun Jul 10 14:12:00 PDT 2016	EXPROT
leave:3:15004	15001	Leave Process	leave	3	XML	PIC	Fri Jul 08 17:04:44 PDT 2016	EXPROT
leave:2:12504	12501	请假流程	leave	2	XML	PIC	Fri Jul 08 14:41:23 PDT 2016	EXPROT
process:4:122507	122504	null	process	4	XML	PIC	Sun Oct 23 15:19:45 PDT 2016	EXPROT

Figure 6.27. Deployed uploaded BPMN process list.

At external BPMN display list, the customer can review a BPMN process file in two ways. The first is to see the process file in XML format (Figure 6.28). Another way is to see the process file as an image (Figure 6.29).

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<definitions xmlns="http://www.omg.org/spec/BPMN/20100218/DCDI" xmlns:xsi="http://www.omg.org/2001/XMLSchema-instance" xmlns:code="http://www.omg.org/2001/XMLSchema" xmlns:activiti="http://activiti.org/bpmn" xmlns:bpmdi="http://www.omg.org/spec/BPMN/20100218/DI"
xmlns:ogdl="http://www.omg.org/spec/OD/20100524/DC" xmlns:ogdi="http://www.omg.org/spec/OD/20100524/DI" typeLanguage="http://www.omg.org/2001/XMLSchema" expressionLanguage="http://www.omg.org/2001/XMLSchema" targetNamespace="http://www.activiti.org/test">
  <process id="Leave" name="Leave Process" isExecutable="true">
    <documentation>Leave Process Display</documentation>
    <startEvent id="startEvent1" name="Start" activiti:initiator="applyUser12"/>
    <userTask id="deptLeaderAudit" name="Dept Leader Approval" activiti:candidateGroups="1-16"/>
    <exclusiveGateway id="exclusiveGateway5" name="Exclusive Gateway"/>
    <userTask id="modifyApply" name="Adjustment" activiti:assignee="japplyuser12"/>
    <extensionElements>
      <activiti:taskListener event="complete" class="com.arvin.controller.AfterModifyApplyContentProcessor"/>
    </extensionElements>
    </userTask>
    <userTask id="hrAudit" name="HR Approval" activiti:candidateGroups="8-17"/>
    <exclusiveGateway id="exclusiveGateway6" name="Exclusive Gateway"/>
    <userTask id="reportBack" name="Cancel Leave" activiti:assignee="japplyuser12"/>
    <extensionElements>
      <activiti:taskListener event="complete" class="com.arvin.controller.ReportBackInProcessor"/>
    </extensionElements>
    </userTask>
    <endEvent id="endEvent1" name="End"/>
    <exclusiveGateway id="exclusiveGateway7" name="Exclusive Gateway"/>
    <sequenceFlow id="Flow0" sourceRef="startEvent1" targetRef="deptLeaderAudit"/>
    <sequenceFlow id="Flow1" sourceRef="deptLeaderAudit" targetRef="exclusiveGateway5"/>
    <sequenceFlow id="Flow2" name="Not Pass" sourceRef="exclusiveGateway5" targetRef="modifyApply"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ ${!DepartmentManagerPass} ]]>
    </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="Flow3" name="Pass" sourceRef="exclusiveGateway5" targetRef="hrAudit"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ ${!DepartmentManagerPass} ]]>
    </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="Flow4" sourceRef="hrAudit" targetRef="exclusiveGateway6"/>
    <sequenceFlow id="Flow5" name="Pas" sourceRef="exclusiveGateway6" targetRef="reportBack"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ ${HRManagerPass} ]]>
    </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="Flow6" sourceRef="reportBack" targetRef="endEvent1"/>
    <sequenceFlow id="Flow7" name="Not Pass" sourceRef="exclusiveGateway6" targetRef="modifyApply"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ ${!HRManagerPass} ]]>
    </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="Flow8" name="Re-Apply" sourceRef="exclusiveGateway7" targetRef="deptLeaderAudit"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ $(reapply) ]]>
    </conditionExpression>
    </sequenceFlow>
    <sequenceFlow id="Flow9" sourceRef="modifyApply" targetRef="exclusiveGateway7"/>
    <sequenceFlow id="Flow10" name="End Process" sourceRef="exclusiveGateway7" targetRef="endEvent1"/>
    <conditionExpression xsi:type="FormalExpression">
      <![CDATA[ $(reapply) ]]>
    </conditionExpression>
    </sequenceFlow>
  </process>
</definitions>
```

Figure 6.28. BPMNN process file in xml format.

This BPMN leave process displays if customers request leaves, the BPMN engine will start the processes and pass the task to department leader. If the department leader accepts these requests, the engine will pass this task to HR department. If the HR department leader also accept these request, then the requests will be sent back customers. If the either HR department leader or department leader does not accept these requirements, the requirements will also be sent back to customers, but customers can either reject these requirements or do some adjustments to restart these requirement processes.

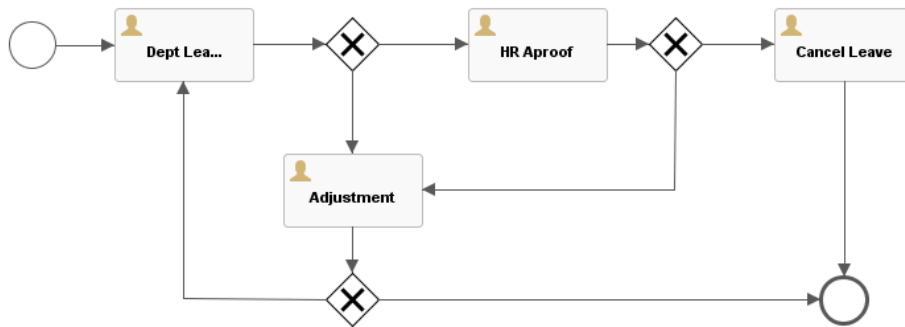


Figure 6.29. BPMNN process displayed by image.

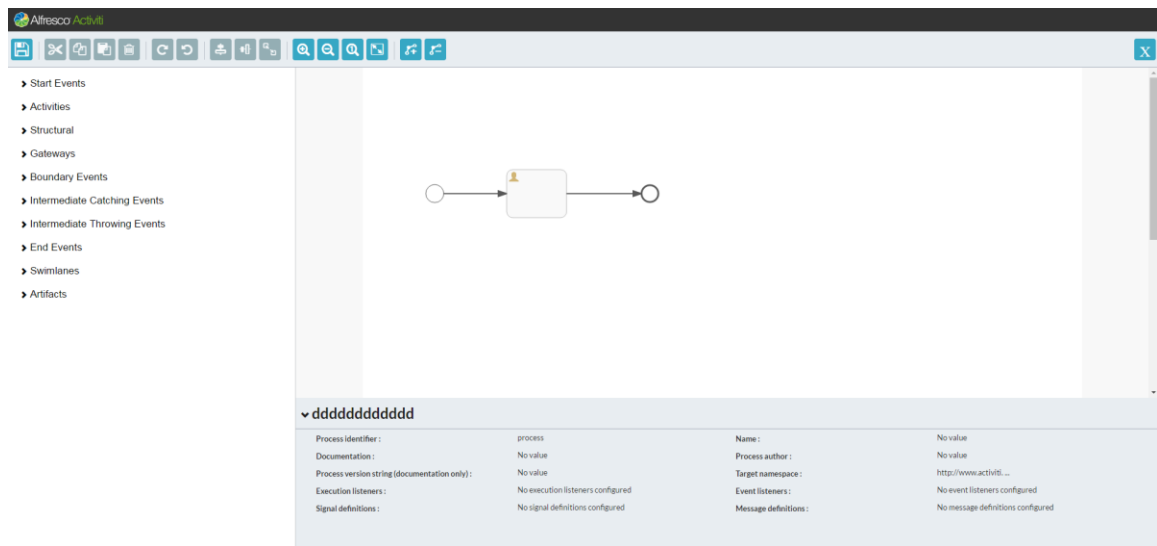


Figure 6.30. Embedded BPMN process editor.

Assign Tasks

This system provides task component which a customer can assign a task to other customers or he/she can also request a task like leave requirement. I design the task component contains two types of task. the first type is BPMN task (Figure 6.31) and another one is the generic task (Figure 6.35 and 3. 36). The BPMN task page has three data lists that they are the current task (Figure 6.31), in process task (Figure 6.32), and finished the task (Figure 6.33).

Current BPMN Task List									
ID	TYPE OF LEAVE	APPLICANT	APPLIED DATA	START DATE	END DATE	CURRENT NODE	CREATE DATE	OPTIONS	
7ad5dd5a-2cb2-426a-be64-74f721e0d99b	General Leave	admin	2016-09-18 00:15:22	2016/09/18 00:15	2016/09/24 00:15	Adjustment	Sun Sep 18 00:15:42 PDT 2016	CONF	
a57e47a2-dd43-46de-a6bf-8df23dc5e2	General Leave	admin	2016-09-18 00:12:17	2016/09/18 00:12	2016/09/21 00:12	Cancel Leave	Sun Oct 16 15:48:19 PDT 2016	CONF	
95e3b3d5-671e-48a8-9056-570400bd100	General Leave	admin	2016-10-23 15:24:29	2016/10/25 15:23	2016/10/28 15:23	Cancel Leave	Sun Oct 23 15:26:36 PDT 2016	CONF	

Figure 6.31. Current BPMN process task list.

Running BPMN Task List									
ID	TYPE OF LEAVE	APPLICANT	APPLIED DATA	START DATE	END DATE	CURRENT NODE	CREATE DATE	OPTIONS	
95e3b3d5-671e-48a8-9056-570400bd100	General Leave	admin	2016-10-23 15:24:29	2016/10/25 15:23	2016/10/28 15:23	Cancel Leave	Sun Oct 23 15:26:36 PDT 2016	CONF	
0a6602b6-90c4-44b9-b78e-89eb8056d52	General Leave	admin	2016-10-16 20:56:25	2016/10/14 20:56	2016/10/27 20:56	Dept Leader Aproof	Sun Oct 16 20:56:25 PDT 2016	CONF	
f1741a3e-e504-4921-b8db-201b5a5e9bad	Emergency Leave	admin	2016-10-16 15:59:06	2016/10/18 15:59	2016/10/21 15:59	HR Aproof	Sun Oct 16 20:57:14 PDT 2016	CONF	
05c25e34-4b2c-42bf-999f-f194637a3f29	General Leave	admin	2016-10-16 15:26:59	2016/10/16 15:26	2016/10/18 15:26	HR Aproof	Sun Oct 16 15:27:46 PDT 2016	CONF	
7ad5dd5a-2cb2-426a-be64-74f721e0d99b	General Leave	admin	2016-09-18 00:15:22	2016/09/18 00:15	2016/09/24 00:15	Adjustment	Sun Sep 18 00:15:42 PDT 2016	CONF	
a57e47a2-dd43-46de-a6bf-8df23dc5e2	General Leave	admin	2016-09-18 00:12:17	2016/09/18 00:12	2016/09/21 00:12	Cancel Leave	Sun Oct 16 15:48:19 PDT 2016	CONF	

Figure 6.32. In process BPMN task list.

Finished BPMN Task List								
☰ Tool Bar ↻ Refresh 👤 Ask a Leave								
ID	TYPE OF LEAVE	APPLICANT	APPLIED DATA	START DATE	END DATE	REAL START DATE	REAL END DATE	
1cb1343e-3deb-493e-a2c9-0ac706429d7c	Sick Leave	admin	2016-10-16 15:34:02	2016/10/16 15:33	2016/10/17 15:33	Tue Oct 18 15:58:00 PDT 2016	Fri Oct 21 15:58:00 PDT 2016	
8eabe9d4-190a-42ab-ac3f-62a9e5ee755b	General Leave	admin	2016-10-16 15:43:32	Sun Oct 16 15:43:00 PDT 2016	Fri Oct 21 15:43:00 PDT 2016	Sun Oct 16 15:49:00 PDT 2016	Mon Oct 17 15:49:00 PDT 2016	

Figure 6.33. Finished BPMN task list.

To add a BPMN task, a customer needs to fill out a BPMN task form (Figure 6.34). The form includes some basic information that will help department leader and HR department leader to determine if he/she needs to reject the request or accept it.



The image shows a form for adding a BPMN task. It features a dark brown background. The form includes the following fields:

- Type of Leave:** A dropdown menu with the selected option being "General Leave".
- Start Date *:** A date input field.
- End Date *:** A date input field.
- Description:** A large text area for entering details.

Figure 6.34. Add BPMN task page.

To assign generic tasks (Figure 6.35), customers should have high-level roles such as department leader. After customers assign generic tasks to other customers. The receivers can find the task at either system tool (Figure 6.36) bar, received task list page (Figure 6.37), or to-do list (Figure 6.38).

The screenshot shows a form for adding a generic task. It contains the following fields:

- Task Type:** A dropdown menu with the text "Please choose a Task Type".
- Task Type:** A dropdown menu with the text "Please choose a User".
- Task Name *:** A text input field containing "Task Name".
- Due Date *:** An empty date input field.
- Process:** A dropdown menu with the text "0 %".
- Task Content:** A large, empty text area for entering task details.

Figure 6.35. Add generic task page.

The screenshot shows a table titled "Received Task List". It includes a toolbar with "Tool Bar" and "Refresh" icons. The table has the following data:

ID	TASK NAME	PROCESS	DUE DATE	ISACCEPT	OPTIONS
1	Test Task	10	2017/01/03 15:28	No	ACCEPT READ

Figure 6.36. Received task list.

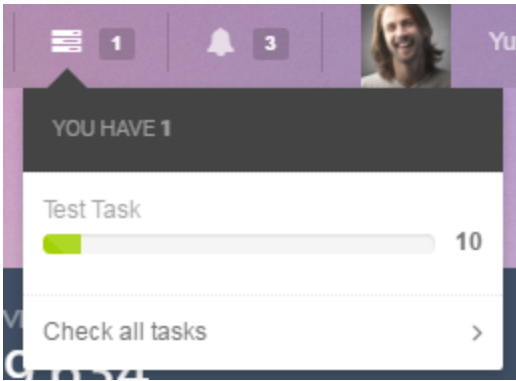


Figure 6.37. System tool bar task list.

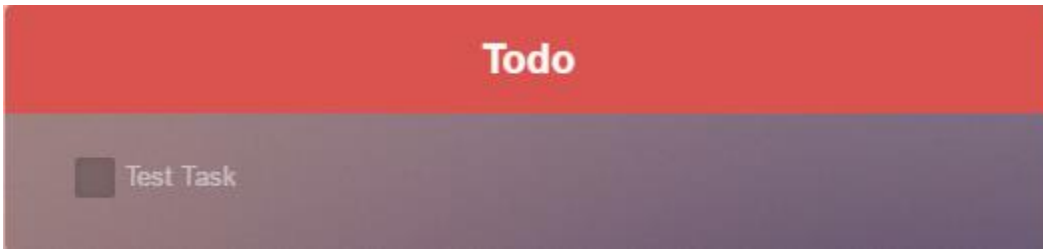


Figure 6.38. To do list of task.

Once a customer finishes a generic task, he/she can click that task to setup the task as completed as Figure 6.39 shows.

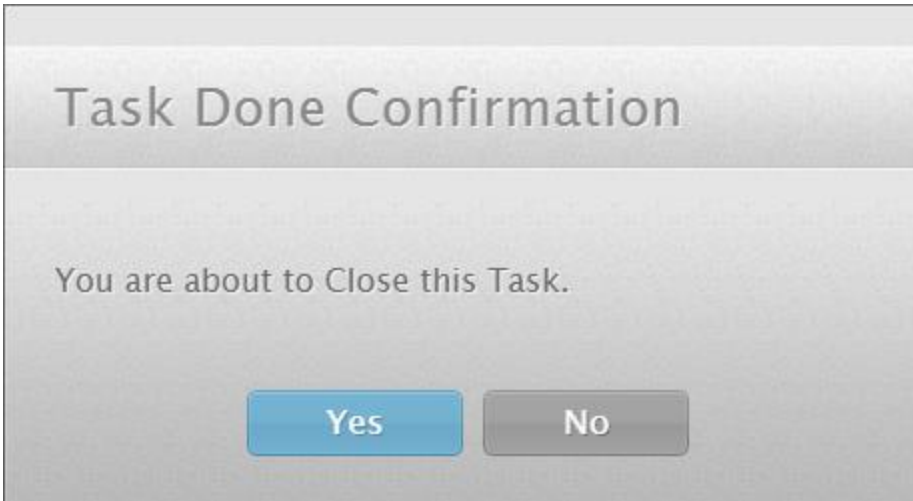
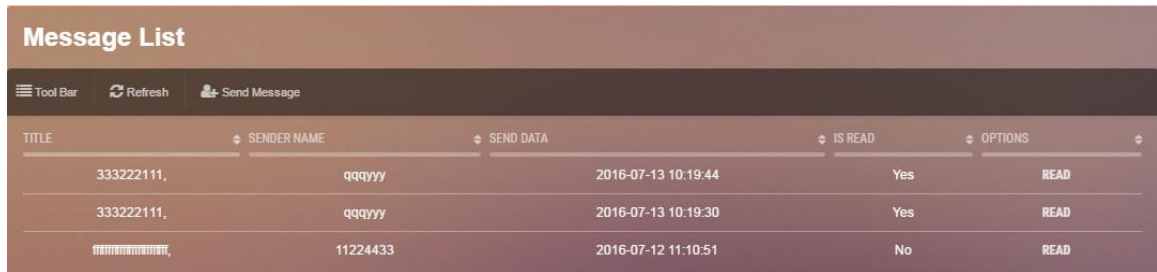


Figure 6.39. Finish a task.

Send Message

The message component allow customers communicate easier with each other by sending an internal message. Then, the receivers are able to find the message at message list page (Figure 6.40) or system toolbar (Figure 6.41).



TITLE	SENDER NAME	SEND DATA	IS READ	OPTIONS
333222111,	qqqqyy	2016-07-13 10:19:44	Yes	READ
333222111,	qqqqyy	2016-07-13 10:19:30	Yes	READ
mmmmmmmmmm,	11224433	2016-07-12 11:10:51	No	READ

Figure 6.40. Message list.

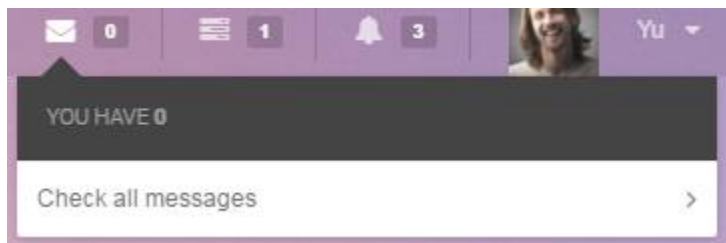


Figure 6.41. System tool bar message list.

CHAPTER SEVEN

CONCLUSION

In the web application generator market of North America, Weebly, WordPress, and Website Builder are widely used web application generators, but these web application generators still cannot generate 100% suitable web application projects for their customers. All the web application generators ask their customers to choose a web template first and then the customers need to modify or add more data to the template at their online editor page. Due to those web application generator, there is no such website page tree diagram which can allow the customers to have an overview of their website constructions. Based on these points, the customers need to know at least how to design and layout their website pages. After the customers have done some adjustments for their websites, the templates can then fit to their business. The project can generate web applications by clicking one button according to the BPMN (Business Process Model Notation) technology. The BPMN engine will become a middle layer that will lead to the logic of page jumping. The customers can modify the content of their web applications through the administration platform with a website page tree diagram. The customers can also manage their web application data at the data section.

This project provides both generic and customized methods to generate web applications. When using the generic method to generate web applications, the customers need to go to the main page and choose one of the existing web

application models and then fill out the template data. By using the customized way to design web applications, the customers should go to the special order page and fill out extra work order forms for the web application support team. Based on the special orders, the support team will pass those work orders to development department. The development department will give suitable development methodologies such as waterfall, agile, and prototype to fit with the customer requirements and develop these requirements. Based on a selected development methodology, customers will receive their required product after one software development life cycle has been implemented.

APPENDIX
CODE OF CRITICAL PARTS

Maven Configuration

```
<developers>
  <developer>
    <id>Arvin</id>
    <name>Yu Zhou</name>
    <email>zhouyuzero@hotmail.com</email>
    <timezone>8</timezone>
  </developer>
</developers>

<properties>
  <java-version>1.8</java-version>
  <jetty.version>8.1.15.v20140411</jetty.version>
  <org.springframework-version>4.0.3.RELEASE</org.springframework-
version>
  <org.aspectj-version>1.7.4</org.aspectj-version>
  <org.slf4j-version>1.7.5</org.slf4j-version>
  <hibernate.version>4.3.5.Final</hibernate.version>
  <jackson.version>2.6.3</jackson.version>
  <activiti.version>5.19.0</activiti.version>
  <guava.version>17.0</guava.version>
  <commons-lang3.version>3.3.2</commons-lang3.version>
  <joda-time.version>2.1</joda-time.version>
  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>

<repositories>
  <repository>
    <id>Activiti</id>

    <url>https://maven.alfresco.com/nexus/content/groups/public</url>
  </repository>
  <repository>
    <id>Maven Central</id>
    <url>http://repo.maven.apache.org/maven2/</url>
  </repository>
  <repository>
    <id>nexus-osc</id>
    <url>http://maven.oschina.net/content/groups/public/</url>
  </repository>
  <repository>
    <id>nexus-osc-thirdparty</id>
    <url>http://maven.oschina.net/content/repositories/thirdparty/</url>
  </repository>
</repositories>
```

```

    </repository>
</repositories>

<build>
    <finalName>${project.artifactId}</finalName>
    <resources>
    <resource>
        <directory>src/main/resources</directory>
        <filtering>true</filtering>
    </resource>
    </resources>
    <plugins>
    <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.1</version>
        <configuration>
            <source>${java-version}</source>
            <target>${java-version}</target>
            <showWarnings>true</showWarnings>
        </configuration>
    </plugin>
    <plugin>
        <groupId>org.apache.tomcat.maven</groupId>
        <artifactId>tomcat7-maven-plugin</artifactId>
        <version>2.2</version>
        <configuration>
            <server>apache-tomcat-7</server>
            <path>/${project.artifactId}</path>
            <url>http://localhost:8080/onlinewebsitegenerator</url>
            <systemProperties>
                <JAVA_OPTS>-Xms1024m -Xmx2048m -XX:PermSize=512m -
                XX:MaxPermSize=1024m -Dfile.encoding=UTF-8</JAVA_OPTS>
            </systemProperties>
        </configuration>
    </plugin>
    <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-war-plugin</artifactId>
        <version>2.4</version>
    </plugin>
    <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-eclipse-plugin</artifactId>

```

```

    <version>2.9</version>
    <configuration>
      <wtpversion>2.0</wtpversion>
      <additionalProjectnatures>

<projectnature>org.springframework.ide.eclipse.core.springnature</projectnature
>
      </additionalProjectnatures>
      <additionalBuildcommands>

<buildcommand>org.springframework.ide.eclipse.core.springbuilder</buildcomm
and>
      </additionalBuildcommands>
      <downloadSources>true</downloadSources>
      <downloadJavadocs>true</downloadJavadocs>
    </configuration>
  </plugin>
  <plugin>
    <groupId>org.mortbay.jetty</groupId>
    <artifactId>jetty-maven-plugin</artifactId>
    <version>${jetty.version}</version>
    <configuration>
      <webApp>
        <contextPath>/${project.artifactId}</contextPath>
      </webApp>
      <stopKey>${project.artifactId}</stopKey>
      <stopPort>9999</stopPort>
    </configuration>
  </plugin>
  <plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-resources-plugin</artifactId>
    <version>2.6</version>
    <configuration>
      <nonFilteredFileExtensions>
        <nonFilteredFileExtension>zip</nonFilteredFileExtension>
        <nonFilteredFileExtension>bar</nonFilteredFileExtension>
        <nonFilteredFileExtension>png</nonFilteredFileExtension>
        <nonFilteredFileExtension>bpmn</nonFilteredFileExtension>
      </nonFilteredFileExtensions>
      <encoding>${project.build.sourceEncoding}</encoding>
    </configuration>
  </plugin>
</plugins>

```

</build>

Web Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" version="2.5">

    <display-name>Online Website Generator Application</display-name>

    <listener>
        <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-class>
        </listener>
    </listener>
    <listener>
        <listener-
class>org.springframework.web.util.Log4jConfigListener</listener-class>
        </listener>
    </listener>
    <listener>
        <listener-
class>com.arvin.sessionlistener.SessionListener</listener-class>
        </listener>

    <session-config>
        <session-timeout>30</session-timeout>
    </session-config>

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
/WEB-INF/spring-dao.xml,
/WEB-INF/spring-service.xml,
/WEB-INF/spring-bpmn.xml
        </param-value>
    </context-param>

    <servlet>
        <servlet-name>appServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-servlet.xml</param-value>
        </init-param>
```

```

        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>appServlet</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>
    <!--
    <servlet>
        <servlet-name>RestletServlet</servlet-name>
    </servlet>
    <servlet>
        <servlet-name>ModelRestServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-modeler.xml</param-
value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>ModelRestServlet</servlet-name>
        <url-pattern>/service/*</url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>RestServlet</servlet-name>
        <servlet-
class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/spring-rest.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>RestServlet</servlet-name>
        <url-pattern>/rest/*</url-pattern>
    </servlet-mapping>

    <context-param>
        <param-name>log4jConfigLocation</param-name>
        <param-value>/WEB-INF/log4j.properties</param-value>
    </context-param>
    <context-param>

```

```

        <param-name>log4jRefreshInterval</param-name>
        <param-value>60000</param-value>
    </context-param>

    <filter>
        <filter-name>CharacterEncodingFilter</filter-name>
        <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
        <init-param>
            <param-name>encoding</param-name>
            <param-value>UTF-8</param-value>
        </init-param>
        <init-param>
            <param-name>forceEncoding</param-name>
            <param-value>true</param-value>
        </init-param>
    </filter>
    <filter-mapping>
        <filter-name>CharacterEncodingFilter</filter-name>
        <url-pattern>/*</url-pattern>
    </filter-mapping>

    <filter>
        <filter-name>springSecurityFilterChain</filter-name>
        <filter-class>
            org.springframework.web.filter.DelegatingFilterProxy
        </filter-class>
    </filter>
    <filter-mapping>
        <filter-name>springSecurityFilterChain</filter-name>
        <url-pattern>/rest/*</url-pattern>
        <dispatcher>ERROR</dispatcher>
        <dispatcher>REQUEST</dispatcher>
    </filter-mapping>

    <filter>
        <filter-name>SessionFilter</filter-name>
        <filter-class>com.arvin.sessionlistener.SessionFilter</filter-class>
    </filter>
    <filter-mapping>
        <filter-name>SessionFilter</filter-name>
        <url-pattern>/*</url-pattern>
        <dispatcher>FORWARD</dispatcher>
        <dispatcher>REQUEST</dispatcher>
    </filter-mapping>

```

```
</filter-mapping>

<error-page>
  <error-code>404</error-code>
  <location>/WEB-INF/pages/page404.jsp</location>
</error-page>
<error-page>
  <error-code>500</error-code>
  <location>/WEB-INF/pages/page500.jsp</location>
</error-page>
</web-app>
```


Spring Configuration

Servlet

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/mvc"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:beans="http://www.springframework.org/schema/beans"
  xmlns:context="http://www.springframework.org/schema/context"
  xsi:schemaLocation="http://www.springframework.org/schema/mvc
    http://www.springframework.org/schema/mvc/spring-mvc.xsd
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context.xsd"
  >
  <annotation-driven/>
  <default-servlet-handler />
  <context:component-scan base-package="com.arvin.controller" />

  <resources location="/resources/" mapping="/resources/**" />
  <resources location="/editor/" mapping="/editor/**"/>
    <resources location="/explorer/" mapping="/explorer/**"/>
    <resources location="/libs/" mapping="/libs/**"/>
    <resources location="/api/" mapping="/api/**"/>
    <resources location="/editor-app/" mapping="/editor-app/**"/>
    <resources location="/diagram-viewer/" mapping="/diagram-viewer/**"/>
    <resources location="/images/" mapping="/images/**"/>

  <beans:bean
class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <beans:property name="prefix" value="/WEB-INF/pages/" />
  <beans:property name="suffix" value=".jsp" />
</beans:bean>

  <beans:bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMultipartResolver"
">
  <!-- setting maximum upload size -->
  <beans:property name="maxUploadSize" value="5000000" />
</beans:bean>
```

```

    <beans:bean id="mappingJacksonHttpMessageConverter"
class="org.springframework.http.converter.json.MappingJackson2HttpMessageC
onverter">
    <beans:property name="supportedMediaTypes">
        <beans:list>
            <beans:value>text/html;charset=UTF-8</beans:value>
        </beans:list>
    </beans:property>
</beans:bean>

<beans:bean
class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandl
erAdapter">
    <beans:property name="messageConverters">
        <beans:list>
            <beans:ref bean="mappingJacksonHttpMessageConverter" />
        </beans:list>
    </beans:property>
</beans:bean>

<!-- Configuring interceptors based on URI -->
<interceptors>
<interceptor>
    <mapping path="/*" />
    <beans:bean
class="com.arvin.interceptor.RequestProcessingTimeInterceptor">
        <beans:property name="excludedUrls">
            <beans:list>
                <beans:value>/</beans:value>
                <beans:value>/customerLogin</beans:value>
                <beans:value>/upload</beans:value>
                <beans:value>/timeout</beans:value>
            </beans:list>
        </beans:property>
    </beans:bean>
</interceptor>
</interceptors>

    <beans:bean
class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandl
erAdapter">
    <beans:property name="messageConverters">
        <beans:list>

```

```

    <beans:bean class =
"org.springframework.http.converter.StringHttpMessageConverter">
    <beans:property name = "supportedMediaTypes">
        <beans:list>
            <beans:value>text/html;charset=UTF-8</beans:value>
        </beans:list>
    </beans:property>
</beans:bean>
</beans:list>
</beans:property>
</beans:bean>

    <beans:bean id="serviceManager"
class="com.arvin.service.manager.ServiceManager">
    <!-- *****Admin***** -->
    <beans:property name="customersService">
        <beans:ref bean="customersService" />
    </beans:property>

    <!-- *****Menu***** -->
    <beans:property name="rolesService">
        <beans:ref bean="rolesService" />
    </beans:property>
    <beans:property name="menusService">
        <beans:ref bean="menusService" />
    </beans:property>

    <!-- *****Task***** -->
    <beans:property name="tasksService">
        <beans:ref bean="tasksService" />
    </beans:property>

    <!-- *****Message***** -->
    <beans:property name="messagesService">
        <beans:ref bean="messagesService" />
    </beans:property>

    <!-- *****System Variables***** -->
    <beans:property name="stateService">
        <beans:ref bean="stateService" />
    </beans:property>
    <beans:property name="countryService">
        <beans:ref bean="countryService" />
    </beans:property>

```

```

<beans:property name="departmentsService">
    <beans:ref bean="departmentsService" />
</beans:property>
<beans:property name="taskTypesService">
    <beans:ref bean="taskTypesService" />
</beans:property>
<beans:property name="pageTypesService">
    <beans:ref bean="pageTypesService" />
</beans:property>
<beans:property name="unitsService">
    <beans:ref bean="unitsService" />
</beans:property>
<beans:property name="systemPortsService">
    <beans:ref bean="systemPortsService" />
</beans:property>
<beans:property name="systemVariablesService">
    <beans:ref bean="systemVariablesService" />
</beans:property>
<beans:property name="variableTypesService">
    <beans:ref bean="variableTypesService" />
</beans:property>

<!-- *****Project***** -->
<beans:property name="projectsService">
    <beans:ref bean="projectsService" />
</beans:property>

<!-- *****Database***** -->
<beans:property name="databasesService">
    <beans:ref bean="databasesService" />
</beans:property>
<beans:property name="databaseTablesService">
    <beans:ref bean="databaseTablesService" />
</beans:property>
<beans:property name="databaseTableVariablesService">
    <beans:ref bean="databaseTableVariablesService" />
</beans:property>

<!-- *****Tomcat***** -->
<beans:property name="tomcatsService">
    <beans:ref bean="tomcatsService" />
</beans:property>
<beans:property name="tomcatFilesService">
    <beans:ref bean="tomcatFilesService" />

```

```

</beans:property>
<beans:property name="tomcatConfsService">
  <beans:ref bean="tomcatConfsService" />
</beans:property>

<!-- *****BPMN***** -->
<beans:property name="act_re_model_Service">
  <beans:ref bean="act_re_model_Service" />
</beans:property>
<beans:property name="act_ge_bytearray_Service">
  <beans:ref bean="act_ge_bytearray_Service" />
</beans:property>
<beans:property name="leavesService">
  <beans:ref bean="leavesService" />
</beans:property>
<beans:property name="bpmnService">
  <beans:ref bean="bpmnService" />
</beans:property>
<beans:property name="act_id_group_Service">
  <beans:ref bean="act_id_group_Service" />
</beans:property>
<beans:property name="act_id_customer_Service">
  <beans:ref bean="act_id_customer_Service" />
</beans:property>

<!-- *****ProjectCatalogs***** -->
<beans:property name="projectCatalogsService">
  <beans:ref bean="projectCatalogsService" />
</beans:property>
</beans:bean>
</beans:beans>

```

Service

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:beans="http://www.springframework.org/schema/beans"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd">

  <!-- ***** Admin ***** -->
  <beans:bean id="customersService"
class="com.arvin.service.impl.CustomersServiceImpl">
    <beans:property name="customersDao"
ref="customersDao"></beans:property>
  </beans:bean>

  <!-- ***** Menu ***** -->
  <beans:bean id="menusService"
class="com.arvin.service.impl.MenusServiceImpl">
    <beans:property name="menusDao"
ref="menusDao"></beans:property>
  </beans:bean>
  <beans:bean id="rolesService"
class="com.arvin.service.impl.RolesServiceImpl">
    <beans:property name="rolesDao"
ref="rolesDao"></beans:property>
  </beans:bean>

  <!-- ***** Task ***** -->
  <beans:bean id="tasksService"
class="com.arvin.service.impl.TasksServiceImpl">
    <beans:property name="tasksDao"
ref="tasksDao"></beans:property>
  </beans:bean>

  <!-- ***** Message ***** -->
  <beans:bean id="messagesService"
class="com.arvin.service.impl.MessagesServiceImpl">
    <beans:property name="messagesDao"
ref="messagesDao"></beans:property>
  </beans:bean>

  <!-- ***** System Variables ***** -->
```

```

        <beans:bean id="stateService"
class="com.arvin.service.impl.StateServiceImpl">
        <beans:property name="stateDao"
ref="stateDao"></beans:property>
        </beans:bean>
        <beans:bean id="countryService"
class="com.arvin.service.impl.CountryServiceImpl">
        <beans:property name="countryDao"
ref="countryDao"></beans:property>
        </beans:bean>
        <beans:bean id="departmentsService"
class="com.arvin.service.impl.DepartmentsServiceImpl">
        <beans:property name="departmentsDao"
ref="departmentsDao"></beans:property>
        </beans:bean>
        <beans:bean id="taskTypesService"
class="com.arvin.service.impl.TaskTypesServiceImpl">
        <beans:property name="taskTypesDao"
ref="taskTypesDao"></beans:property>
        </beans:bean>
        <beans:bean id="unitsService"
class="com.arvin.service.impl.UnitsServiceImpl">
        <beans:property name="unitsDao"
ref="unitsDao"></beans:property>
        </beans:bean>
        <beans:bean id="pageTypesService"
class="com.arvin.service.impl.PageTypesServiceImpl">
        <beans:property name="pageTypesDao"
ref="pageTypesDao"></beans:property>
        </beans:bean>
        <beans:bean id="systemPortsService"
class="com.arvin.service.impl.SystemPortsServiceImpl">
        <beans:property name="systemPortsDao"
ref="systemPortsDao"></beans:property>
        </beans:bean>
        <beans:bean id="systemVariablesService"
class="com.arvin.service.impl.SystemVariablesServiceImpl">
        <beans:property name="systemVariablesDao"
ref="systemVariablesDao"></beans:property>
        </beans:bean>
        <beans:bean id="variableTypesService"
class="com.arvin.service.impl.VariableTypesServiceImpl">
        <beans:property name="variableTypesDao"
ref="variableTypesDao"></beans:property>

```

```

    </beans:bean>

    <!-- *****Project***** -->
    <beans:bean id="projectsService"
class="com.arvin.service.impl.ProjectsServiceImpl">
        <beans:property name="projectsDao"
ref="projectsDao"></beans:property>
    </beans:bean>

    <!-- *****Tomcat***** -->
    <beans:bean id="databasesService"
class="com.arvin.service.impl.DatabasesServiceImpl">
        <beans:property name="databasesDao"
ref="databasesDao"></beans:property>
    </beans:bean>
    <beans:bean id="databaseTablesService"
class="com.arvin.service.impl.DatabaseTablesServiceImpl">
        <beans:property name="databaseTablesDao"
ref="databaseTablesDao"></beans:property>
    </beans:bean>
    <beans:bean id="databaseTableVariablesService"
class="com.arvin.service.impl.DatabaseTableVariablesServiceImpl">
        <beans:property name="databaseTableVariablesDao"
ref="databaseTableVariablesDao"></beans:property>
    </beans:bean>

    <!-- *****Database***** -->
    <beans:bean id="tomcatsService"
class="com.arvin.service.impl.TomcatsServiceImpl">
        <beans:property name="tomcatsDao"
ref="tomcatsDao"></beans:property>
    </beans:bean>
    <beans:bean id="tomcatFilesService"
class="com.arvin.service.impl.TomcatFilesServiceImpl">
        <beans:property name="tomcatFilesDao"
ref="tomcatFilesDao"></beans:property>
    </beans:bean>
    <beans:bean id="tomcatConfsService"
class="com.arvin.service.impl.TomcatConfsServiceImpl">
        <beans:property name="tomcatConfsDao"
ref="tomcatConfsDao"></beans:property>
    </beans:bean>

    <!-- *****BPMN***** -->

```



```

        <beans:bean id="act_re_model_Service"
class="com.arvin.service.impl.Act_re_model_Service_Impl">
            <beans:property name="act_re_model_Dao"
ref="act_re_model_Dao"></beans:property>
        </beans:bean>
        <beans:bean id="act_ge_bytearray_Service"
class="com.arvin.service.impl.Act_ge_bytearray_Service_Impl">
            <beans:property name="act_ge_bytearray_Dao"
ref="act_ge_bytearray_Dao"></beans:property>
        </beans:bean>
        <beans:bean id="leavesService"
class="com.arvin.service.impl.LeavesServiceImpl">
            <beans:property name="leavesDao"
ref="leavesDao"></beans:property>
        </beans:bean>
        <beans:bean id="bpmnService"
class="com.arvin.service.impl.BpmnServiceImpl">
            <beans:property name="bpmnDao"
ref="bpmnDao"></beans:property>
        </beans:bean>
        <beans:bean id="act_id_customer_Service"
class="com.arvin.service.impl.Act_id_customer_Service_Impl">
            <beans:property name="act_id_customer_Dao"
ref="act_id_customer_Dao"></beans:property>
        </beans:bean>
        <beans:bean id="act_id_group_Service"
class="com.arvin.service.impl.Act_id_group_Service_Impl">
            <beans:property name="act_id_group_Dao"
ref="act_id_group_Dao"></beans:property>
        </beans:bean>

        <!-- *****ProjectCatalogs***** -->
        <beans:bean id="projectCatalogsService"
class="com.arvin.service.impl.ProjectCatalogsServiceImpl">
            <beans:property name="projectCatalogsDao"
ref="projectCatalogsDao"></beans:property>
        </beans:bean>
    </beans>

```

BPMN Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xsi:schemaLocation="
           http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans-4.1.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd">

    <context:component-scan base-
package="org.activiti.conf,org.activiti.rest.editor,org.activiti.rest.service">
        <context:exclude-filter type="annotation"
expression="org.springframework.stereotype.Controller"/>
    </context:component-scan>

    <bean id="objectMapper"
class="com.fasterxml.jackson.databind.ObjectMapper"/>

    <bean id="uuidGenerator"
class="org.activiti.engine.impl.persistence.StrongUuidGenerator" />

    <bean id="processEngine"
class="org.activiti.spring.ProcessEngineFactoryBean">
        <property name="processEngineConfiguration"
ref="processEngineConfiguration" />
    </bean>

    <bean id="processEngineConfiguration"
class="org.activiti.spring.SpringProcessEngineConfiguration">
        <property name="dataSource" ref="dataSource" />
        <property name="databaseSchemaUpdate" value="true" />
        <property name="jobExecutorActivate" value="false"/>
        <!--<property name="history" value="full"/>-->
        <property name="transactionManager" ref="transactionManager" />
        <property name="processDefinitionCacheLimit" value="10"/>
    </bean>

    <bean id="repositoryService" factory-bean="processEngine" factory-
method="getRepositoryService" />
    <bean id="runtimeService" factory-bean="processEngine" factory-
method="getRuntimeService" />

```

```
<bean id="taskService" factory-bean="processEngine" factory-  
method="getTaskService" />  
<bean id="historyService" factory-bean="processEngine" factory-  
method="getHistoryService" />  
<bean id="managementService" factory-bean="processEngine" factory-  
method="getManagementService" />  
<bean id="IdentityService" factory-bean="processEngine" factory-  
method="getIdentityService" />  
<bean id="formService" factory-bean="processEngine" factory-  
method="getFormService" />  
  
<bean id="restResponseFactory"  
class="org.activiti.rest.service.api.RestResponseFactory" />  
<bean id="contentTypeResolver"  
class="org.activiti.rest.common.application.DefaultContentTypeResolver" />  
</beans>
```

Hibernate Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:beans="http://www.springframework.org/schema/beans"
       xmlns:tx="http://www.springframework.org/schema/tx"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd"
       >
    <tx:annotation-driven transaction-manager="transactionManager" proxy-
target-class="true"/>

    <beans:bean id="propertyConfigurer"
class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer"
>
        <beans:property name="location" value="/WEB-
INF/database.properties"/>
    </beans:bean>

    <beans:bean id="dataSource"
class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
        <!-- Connection Info -->
        <beans:property name="driverClassName"
value="{jdbc.driverClassName}" />
        <beans:property name="url" value="{jdbc.url}" />
        <beans:property name="customername" value="{jdbc.customername}" />
        <beans:property name="password" value="{jdbc.password}" />
        <!-- Connection Pooling Info -->
        <beans:property name="maxActive" value="{dbcp.maxActive}"/>
        <beans:property name="maxIdle" value="{dbcp.maxIdle}"/>
        <!-- Idle -->
        <beans:property name="defaultAutoCommit" value="false"/>
        <beans:property name="timeBetweenEvictionRunsMillis"
value="3600000"/>
        <beans:property name="minEvictableIdleTimeMillis" value="3600000"/>
    </beans:bean>

    <beans:bean id="transactionManager"
class="org.springframework.orm.hibernate4.HibernateTransactionManager">
        <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />

```

```

</beans:bean>

<beans:bean id="hibernate4AnnotatedSessionFactory"
class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">
  <beans:property name="dataSource" ref="dataSource" />
  <beans:property name="annotatedClasses">
    <beans:list>
      <beans:value>com.arvin.entity.Customers</beans:value>
      <beans:value>com.arvin.entity.Roles</beans:value>
      <beans:value>com.arvin.entity.Customers</beans:value>
      <beans:value>com.arvin.entity.State</beans:value>
      <beans:value>com.arvin.entity.Country</beans:value>
      <beans:value>com.arvin.entity.Tasks</beans:value>
      <beans:value>com.arvin.entity.TaskTypes</beans:value>
      <beans:value>com.arvin.entity.Messages</beans:value>
      <beans:value>com.arvin.entity.Departments</beans:value>
      <beans:value>com.arvin.entity.Menus</beans:value>
      <beans:value>com.arvin.entity.Authorities</beans:value>
      <beans:value>com.arvin.entity.Units</beans:value>
      <beans:value>com.arvin.entity.Projects</beans:value>
      <beans:value>com.arvin.entity.Databases</beans:value>
      <beans:value>com.arvin.entity.DatabaseTables</beans:value>

<beans:value>com.arvin.entity.DatabaseTableVariables</beans:value>
      <beans:value>com.arvin.entity.Templates</beans:value>
      <beans:value>com.arvin.entity.TomcatConfs</beans:value>
      <beans:value>com.arvin.entity.TomcatFiles</beans:value>
      <beans:value>com.arvin.entity.Tomcats</beans:value>
      <beans:value>com.arvin.entity.VariableTypes</beans:value>
      <beans:value>com.arvin.entity.PageTypes</beans:value>
      <beans:value>com.arvin.entity.SystemVariables</beans:value>
      <beans:value>com.arvin.entity.SystemPorts</beans:value>
      <beans:value>com.arvin.entity.ProjectCatalogs</beans:value>
      <beans:value>com.arvin.entity.activiti.Act_re_model</beans:value>

<beans:value>com.arvin.entity.activiti.Act_ge_bytearray</beans:value>
      <beans:value>com.arvin.entity.BpmnProcessFiles</beans:value>
    </beans:list>
  </beans:property>
  <beans:property name="hibernateProperties">
    <beans:props>
      <beans:prop
key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</beans:prop>

```

```

        <beans:prop
key="hibernate.ejb.naming_strategy">org.hibernate.cfg.ImprovedNamingStrateg
y</beans:prop>
        <beans:prop
key="hibernate.show_sql">${hibernate.show_sql}</beans:prop>
        <beans:prop
key="hibernate.dialect">${hibernate.dialect}</beans:prop>
        <beans:prop key="hibernate.format_sql">>true</beans:prop>
        <beans:prop key="hibernate.hbm2ddl.auto">update</beans:prop>
        <beans:prop
key="hibernate.connection.useUnicode">>true</beans:prop>
        <beans:prop key="hibernate.connection.characterEncoding">UTF-
8</beans:prop>
        <beans:prop key="hibernate.max_fetch_depth">0</beans:prop>
        <beans:prop
key="hibernate.jdbc.use_scrollable_resultset">>true</beans:prop>
        <beans:prop key="hibernate.jdbc.fetch_size">30</beans:prop>
        <beans:prop
key="hibernate.jdbc.batch_size">20</beans:prop>
        <beans:prop
key="hibernate.cache.use_second_level_cache">${hibernate.cache.use_second
_level_cache}</beans:prop>
        <beans:prop
key="hibernate.cache.use_query_cache">${hibernate.cache.use_query_cache}<
/beans:prop>
        <beans:prop
key="hibernate.cache.region.factory_class">${hibernate.cache.region.factory_cla
ss}</beans:prop>
        <beans:prop
key="hibernate.cache.provider_configuration_file_resource_path">${hibernate.ca
che.provider_configuration_file_resource_path}</beans:prop>
    </beans:props>
</beans:property>
</beans:bean>

    <beans:bean id="transactionInterceptor"
class="org.springframework.transaction.interceptor.TransactionInterceptor">
        <beans:property name="transactionManager">
            <beans:ref bean="transactionManager" />
        </beans:property>
        <beans:property name="transactionAttributes">
            <beans:props>
                <beans:prop
key="delete*">PROPAGATION_REQUIRED</beans:prop>

```

```

        <beans:prop
key="operate*">PROPAGATION_REQUIRED,-Exception</beans:prop>
        <beans:prop
key="insert*">PROPAGATION_REQUIRED,-Exception</beans:prop>
        <beans:prop
key="update*">PROPAGATION_REQUIRED,-Exception</beans:prop>
        <beans:prop
key="save*">PROPAGATION_REQUIRED</beans:prop>
        <beans:prop
key="find*">PROPAGATION_REQUIRED,readonly</beans:prop>
        </beans:props>
    </beans:property>
</beans:bean>

    <beans:bean id="txProxy"
class="org.springframework.aop.framework.autoproxy.BeanNameAutoProxyCrea
tor">
        <beans:property name="beanNames">
            <beans:list>
                <beans:value>*Service*</beans:value>
            </beans:list>
        </beans:property>
        <beans:property name="interceptorNames">
            <beans:list>
                <beans:value>transactionInterceptor</beans:value>
            </beans:list>
        </beans:property>
    </beans:bean>

    <!-- *****Admin***** -->
    <beans:bean id="customersDao"
class="com.arvin.dao.impl.CustomersDaoImpl">
        <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>

    <!-- *****Message***** -->
    <beans:bean id="messagesDao"
class="com.arvin.dao.impl.MessagesDaoImpl">
        <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>

    <!-- *****Task***** -->

```

```

        <beans:bean id="tasksDao" class="com.arvin.dao.impl.TasksDaoImpl">
            <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
        </beans:bean>

<!-- *****Menu***** -->
<beans:bean id="menusDao" class="com.arvin.dao.impl.MenusDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="rolesDao" class="com.arvin.dao.impl.RolesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>

<!-- *****System Variables***** -->
<beans:bean id="countryDao" class="com.arvin.dao.impl.CountryDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="stateDao" class="com.arvin.dao.impl.StateDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="departmentsDao"
class="com.arvin.dao.impl.DepartmentsDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="taskTypesDao"
class="com.arvin.dao.impl.TaskTypesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="unitsDao" class="com.arvin.dao.impl.UnitsDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
<beans:bean id="systemPortsDao"
class="com.arvin.dao.impl.SystemPortsDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>

```



```

    <beans:bean id="systemVariablesDao"
class="com.arvin.dao.impl.SystemVariablesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>
    <beans:bean id="variableTypesDao"
class="com.arvin.dao.impl.VariableTypeDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>
    <beans:bean id="pageTypesDao"
class="com.arvin.dao.impl.PageTypesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>

<!-- *****Project***** -->
    <beans:bean id="projectsDao"
class="com.arvin.dao.impl.ProjectsDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>

<!-- *****Database***** -->
    <beans:bean id="databasesDao"
class="com.arvin.dao.impl.DatabasesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>
    <beans:bean id="databaseTablesDao"
class="com.arvin.dao.impl.DatabaseTablesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>
    <beans:bean id="databaseTableVariablesDao"
class="com.arvin.dao.impl.DatabaseTableVariablesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
    </beans:bean>

<!-- *****Tomcat***** -->
    <beans:bean id="tomcatsDao"
class="com.arvin.dao.impl.TomcatsDaoImpl">

```

```

    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="tomcatFilesDao"
class="com.arvin.dao.impl.TomcatFilesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="tomcatConfsDao"
class="com.arvin.dao.impl.TomcatConfsDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>

<!-- *****BPMN***** -->
  <beans:bean id="act_re_model_Dao"
class="com.arvin.dao.impl.Act_re_model_Dao_Impl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="act_ge_bytearray_Dao"
class="com.arvin.dao.impl.Act_ge_bytearray_Dao_Impl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="leavesDao" class="com.arvin.dao.impl.LeavesDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="bpmnDao" class="com.arvin.dao.impl.BpmnDaoImpl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="act_id_group_Dao"
class="com.arvin.dao.impl.Act_id_group_Dao_Impl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>
  <beans:bean id="act_id_customer_Dao"
class="com.arvin.dao.impl.Act_id_customer_Dao_Impl">
    <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
  </beans:bean>

```

```
<!-- *****ProjectCatalogs***** -->
<beans:bean id="projectCatalogsDao"
class="com.arvin.dao.impl.ProjectCatalogsDaoImpl">
  <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
</beans:bean>
</beans>
```

MVC Controller

```
package com.arvin.controller;

import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.TreeMap;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;

import com.arvin.entity.Country;
import com.arvin.entity.Databases;
import com.arvin.entity.Departments;
import com.arvin.entity.Menus;
import com.arvin.entity.PageTypes;
import com.arvin.entity.Roles;
import com.arvin.entity.State;
import com.arvin.entity.SystemPorts;
import com.arvin.entity.SystemVariables;
import com.arvin.entity.TaskTypes;
import com.arvin.entity.Tasks;
import com.arvin.entity.Units;
import com.arvin.entity.Customers;
import com.arvin.entity.VariableTypes;
import com.arvin.entity.view.MenusView;
import com.arvin.service.manager.ServiceManager;

@Controller
public class BaseController {
    public static final Logger logger =
    LoggerFactory.getLogger(BaseController.class);

    public ServiceManager serviceManager;

    @Autowired(required = true)
    @Qualifier(value = "serviceManager")
    public void setServiceManager(ServiceManager serviceManager) {
        this.serviceManager = serviceManager;
    }
}
```

```

    public Map<String, String> getVariableTypeList(){
        List<VariableTypes> variableTypeList =
    serviceManager.getVariableTypesService().getVariableTypeList();
        Map<String, String > variableTypes = new HashMap<String, String>();
        for(VariableTypes v : variableTypeList){
            variableTypes.put(v.getVariableTypeName(),
    v.getVariableTypeName());
        }
        return variableTypes;
    }

    public Map<String, String> getSystemPortList(){
        List<SystemPorts> res =
    serviceManager.getSystemPortsService().getTop10SystemPortList();
        Map<String, String > systemPortList = new HashMap<String, String>();
        for(SystemPorts s : res){
            systemPortList.put(s.getPort(), s.getPort());
        }
        return systemPortList;
    }

    public Map<Integer, String> getDatabaseList(){
        List<Databases> databasesList =
    serviceManager.getDatabasesService().getDatabasesList();
        Map<Integer, String > databases = new HashMap<Integer, String>();
        for(Databases d : databasesList){
            databases.put(d.getId(), d.getDatabaseName());
        }
        return databases;
    }

    public Map<Integer, String> getUnitList(){
        List<Units> unitList = serviceManager.getUnitsService().getUnitList();
        Map<Integer, String > units = new HashMap<Integer, String>();
        for(Units u : unitList){
            units.put(u.getId(), u.getUnitName());
        }
        return units;
    }

    public Map<String, String> getPageTypeList(){
        List<PageTypes> pageTypeList =
    serviceManager.getPageTypesService().getPageTypeList();

```

```

        Map<String, String > pageTypes = new HashMap<String, String>();
        for(PageTypes pt : pageTypeList){
            pageTypes.put(pt.getPageTypeName(), pt.getPageTypeName());
        }
        return pageTypes;
    }

    public Map<Integer, String> getIntPageTypeList(){
        List<PageTypes> pageTypeList =
serviceManager.getPageTypesService().getPageTypeList();
        Map<Integer, String > pageTypes = new HashMap<Integer, String>();
        for(PageTypes pt : pageTypeList){
            pageTypes.put(pt.getId(), pt.getPageTypeName());
        }
        return pageTypes;
    }

    public Map<Integer, String> getTaskTypeList(){
        List<TaskTypes> taskTypeList =
serviceManager.getTaskTypesService().getTaskTypeList();
        Map<Integer, String > taskTypes = new HashMap<Integer, String>();
        for(TaskTypes t : taskTypeList){
            taskTypes.put(t.getId(), t.getTaskTypeName());
        }
        return taskTypes;
    }

    public Map<String, String> getDepartmentCustomers(int departmentId,int
positionLevel){
        List<Customers> list =
serviceManager.getCustomersService().getCustomersByDepartmentId(departme
ntId,positionLevel);
        Map<String, String > departments = new HashMap<String, String>();
        for(Customers u : list){
            departments.put(u.getUid(), u.getLastName()+",
"+u.getFirstName());
        }
        return departments;
    }

    public Map<String, String> getCustomers(){
        List<Customers> list =
serviceManager.getCustomersService().getCustomerList();
        Map<String, String > customers = new HashMap<String, String>();

```

```

        for(Customers u : list){
            customers.put(u.getUid(), u.getLastName()+" "+u.getFirstName());
        }
        return customers;
    }

    public Map<String, String> getDepartmentCustomerList(int rid){
        List<Customers> list =
serviceManager.getCustomersService().getCustomersByRoleId(rid);
        Map<String, String > departments = new HashMap<String, String>();
        for(Customers u : list){
            departments.put(u.getUid(), u.getFirstName()+"
"+u.getLastName());
        }
        return departments;
    }

    public Map<Integer, String> getDepartmentList(){
        List<Departments> departmentList =
serviceManager.getDepartmentsService().getDepartmentList();
        Map<Integer, String > departments = new HashMap<Integer, String>();
        for(Departments d : departmentList){
            departments.put(d.getId(), d.getDepartmentName());
        }
        return departments;
    }

    public Map<Integer, String> getRoleList(){
        List<Roles> roleList = serviceManager.getRolesService().getRoleList();
        Map<Integer, String > roles = new HashMap<Integer, String>();
        for(Roles r : roleList){
            roles.put(r.getId(), r.getRoleName());
        }
        return roles;
    }

    public Map<Integer, String> getCountryList(){
        List<Country> countryList =
serviceManager.getCountryService().getCountryList();
        Map<Integer, String > country = new HashMap<Integer, String>();
        for(Country c : countryList){
            country.put(c.getId(), c.getCountryName());
        }
        return country;
    }

```

```

}

public Map<Integer, String> getStateList(){
    List<State> stateList = serviceManager.getStateService().getStateList();
    Map<Integer, String > state = new HashMap<Integer, String>();
    for(State s : stateList){
        state.put(s.getId(), s.getStateName());
    }
    return state;
}

public Map<Integer, String> getMenuRoot(){
    List<Menus> menuList =
serviceManager.getMenusService().getMenusRoot();
    Map<Integer, String > menuRoot = new HashMap<Integer, String>();
    for(Menus m : menuList){
        menuRoot.put(m.getId(), m.getMenuName());
    }
    return menuRoot;
}

public Map<Integer, String> getPositionLevel(){
    Map<Integer, String > positionLevel = new HashMap<Integer, String>();
    for(int i=1; i<6; i++){
        positionLevel.put(i, "Level "+i);
    }
    return positionLevel;
}

public Map<Integer, String> getProcessLevel(){
    Map<Integer, String > positionLevel = new HashMap<Integer, String>();
    for(int i=10; i<=100; i+=10){
        positionLevel.put(i, i+" %");
    }
    Map<Integer, String > sortMap = new TreeMap<Integer,
String >(positionLevel);
    return sortMap;
}

public List<MenuView> getCustomerMenusRoot(int rid){
    List<MenuView> list =
serviceManager.getMenusService().getMenusRootByRid(rid);
    return list;
}

```



```
    public List<MenuView> getCustomerMenuLeaf(int rid){
        List<MenuView> list =
serviceManager.getMenuService().getMenuLeafByRid(rid);
        return list;
    }

    public List<Tasks> getTopFiveTaskListByReceiverId(String uid){
        List<Tasks> list =
serviceManager.getTasksService().getTopFiveTaskListByReceiverId(uid);
        return list;
    }

    public SystemVariables getSystemVariables(){
        SystemVariables systemVariables =
serviceManager.getSystemVariablesService().findSystemVariables();
        return systemVariables;
    }
}
```

Java Data Reflection

```
package com.arvin.util;

import java.lang.reflect.Field;
import java.sql.Blob;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Map;

public class ObjectListToModelList<T> {
    private final Class<T> clazz;

    public ObjectListToModelList(Class<T> clazz) {
        this.clazz = clazz;
    }

    public List<String> getPropertyNames(Object obj) {
        List<String> list = new ArrayList<String>();
        Field[] fs = obj.getClass().getDeclaredFields();
        for (Field f : fs) {
            list.add(f.getName());
        }
        return list;
    }

    public List<T> getModelList(@SuppressWarnings("rawtypes") List<Map>
res) throws SecurityException, NoSuchFieldException,
InstantiationException, IllegalAccessException {
        List<T> list = new ArrayList<T>();
        for (Map<?, ?> m : res) {
            T tempClass = (T) clazz.newInstance();
            List<?> propertyNames = this.getPropertyNames(tempClass);
            for (int i = 0; i < propertyNames.size(); i++) {
                Field f =
tempClass.getClass().getDeclaredField(propertyNames.get(i).toString());
                f.setAccessible(true);
                try {
                    if
(f.getType().getName().equals("java.util.Date")) {
                        SimpleDateFormat formatter = new
SimpleDateFormat("yyyy-MM-dd");
```

```

                Date d =
formatter.parse(m.get(propertyName.get(i).toString().trim()).toString());
                f.set(tempClass, d);
            }else if
(f.getType().getName().equals("java.lang.Integer") &&
m.get(propertyName.get(i).toString().trim())==null) {
                f.set(tempClass, 0);
            }else if (f.getType().getName().equals("byte"))
{
                Blob blob = (Blob)
m.get(propertyName.get(i));
                long blobLength = blob.length();
                int pos = 1;
                int len = 10;
                byte[] bytes = blob.getBytes(pos, len);
                f.set(tempClass, bytes);
            }else if (f.getType().getName().equals("int") &&
m.get(propertyName.get(i).toString().trim())==null) {
                f.set(tempClass, 0);
            }else if
(f.getType().getName().equals("java.lang.String") &&
m.get(propertyName.get(i).toString().trim())==null) {
                f.set(tempClass, "");
            }else {
                if(f.getName().equals("id") &&
f.getType().getName().equals("int")){
                    f.set(tempClass,Integer.parseInt(m.get(propertyName.get(i).toString().trim(
)).toString()));
                }else{
                    f.set(tempClass,m.get(propertyName.get(i).toString().trim()));
                }
            }
        } catch (Exception e) {
            System.out.println("The Value
is:"+m.get(propertyName.get(i).toString().trim()));
            System.out.println(propertyName.get(i).toString().trim()+"["+f.getType().ge
tName()+"]" + "=>setValue() error!");
        }
    }
    list.add((T) tempClass);
}

```

```
    return list;  
  }  
}
```

Model

```
package com.arvin.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="country")
public class Country {
    @Id
    @Column(name="id")
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private int id;
    private String countryName;
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getCountryName() {
        return countryName;
    }
}
```

```
    }  
    public void setCountryName(String countryName) {  
        this.countryName = countryName;  
    }  
}
```

DAO

```
package com.arvin.dao.impl;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.transform.Transformers;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;

import com.arvin.dao.CountryDao;
import com.arvin.entity.Country;
import com.arvin.entity.Page;
import com.arvin.util.ObjectListToModelList;

@Repository
public class CountryDaoImpl implements CountryDao{

    private static final Logger logger =
    LoggerFactory.getLogger(CountryDaoImpl.class);

    @Autowired
    private SessionFactory sessionFactory;

    public void setSessionFactory(SessionFactory sessionFactory) {
        this.sessionFactory = sessionFactory;
    }

    @Transactional
    public void addCountry(Country country) {
        // TODO Auto-generated method stub
        Session session = this.sessionFactory.getCurrentSession();
        session.persist(country);
        logger.info("Country saved successfully, Country Details="+country);
    }
}
```

```

@Transactional
    public void updateCountry(Country country) {
        // TODO Auto-generated method stub
        Session session = this.sessionFactory.getCurrentSession();
        session.update(country);
        logger.info("Country updated successfully, Country Details="+country);
    }

```

```

@Transactional
    public void deleteCountryById(int id) {
        // TODO Auto-generated method stub
        Session session = this.sessionFactory.getCurrentSession();
        Country c = (Country) session.load(Country.class, new Integer(id));
        if(null != c){
            session.delete(c);
        }
    }

```

```

@Transactional
    public List<Country> getCountryListByPage(Page page) {
        // TODO Auto-generated method stub
        String sql="",tsql="";
        sql="select * from Country limit
"+page.getPageIndex()+",""+page.getPageSize();
        tsql="select count(id) from Country";
        Session session = this.sessionFactory.getCurrentSession();
        Query query=session.createQuery(tsql);
        page.setTotalCount(Integer.parseInt(query.uniqueResult().toString()));
        query = session.createQuery(sql);
        query.setResultTransformer(Transformers.ALIAS_TO_ENTITY_MAP);
        List<Map> res = query.list();
        ObjectListToModelList<Country> otml=new
ObjectListToModelList<Country>(Country.class);
        List<Country> back=new ArrayList<Country>();
        try {
            back=otml.getModelList(res);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return back;
    }

```

```

@Transactional
    public Country getCountryById(int id) {

```



```

        // TODO Auto-generated method stub
        Session session = this.sessionFactory.getCurrentSession();
        Country c = (Country) session.get(Country.class, new Integer(id));
    return c;
    }

    @Transactional
    public List<Country> getCountryList() {
        // TODO Auto-generated method stub
        Session session = this.sessionFactory.getCurrentSession();
        List<Country> countryList = session.createQuery("from Country").list();
        for(Country c : countryList){
            logger.info("Country List: "+c);
        }
        return countryList;
    }

    @Transactional
    public boolean findCountryByName(String countryName) {
        // TODO Auto-generated method stub
        List<Country> countryList = new ArrayList<Country>();
        Query query = this.sessionFactory.getCurrentSession().createQuery("from
Country c where c.countryName = :countryName");
        query.setParameter("countryName", countryName);
        countryList = query.list();
        if (countryList.size() > 0){
            return false;
        }
        return true;
    }
}

```

Service

```
package com.arvin.service.impl;

import java.util.List;

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.arvin.dao.CountryDao;
import com.arvin.entity.Country;
import com.arvin.entity.Page;
import com.arvin.service.CountryService;

@Service
public class CountryServiceImpl implements CountryService {

    private CountryDao countryDao;

    public void setCountryDao(CountryDao countryDao) {
        this.countryDao = countryDao;
    }

    @Transactional
    public void addCountry(Country country) {
        // TODO Auto-generated method stub
        countryDao.addCountry(country);
    }

    @Transactional
    public void updateCountry(Country country) {
        // TODO Auto-generated method stub
        countryDao.updateCountry(country);
    }

    @Transactional
    public void deleteCountryById(int id) {
        // TODO Auto-generated method stub
        countryDao.deleteCountryById(id);
    }

    @Transactional
    public List<Country> getCountryListByPage(Page page) {
        // TODO Auto-generated method stub
    }
}
```

```

        return countryDao.getCountryListByPage(page);
    }

    @Transactional
    public Country getCountryById(int id) {
        // TODO Auto-generated method stub
        return countryDao.getCountryById(id);
    }

    @Transactional
    public List<Country> getCountryList() {
        // TODO Auto-generated method stub
        return countryDao.getCountryList();
    }

    @Override
    public boolean findCountryByName(String countryName) {
        // TODO Auto-generated method stub
        return countryDao.findCountryByName(countryName);
    }
}

```

REFERENCES

- [1] A. Leonard, *JBoss Tools 3 Developers Guide*, Packet Publishing Ltd, April 2009.
- [2] B. Laurie and P. Laurie, *Apache the definitive Guide 3rd*, O'REILLY, Dec. 2002.
- [3] B. Rucker, *Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve, and Automate Processes in Your Company*, 2 edition, CreateSpace Independent Publishing Platform, Dec. 5, 2014.
- [4] B. Silver, *Bpmn Method and Style, 2nd Edition, with Bpmn Implementer's Guide: A Structured Approach for Business Process Modeling and Implementation Using Bpmn 2*, Cody-Cassidy Press, Oct. 17, 2011.
- [5] G. Zambon and M. Sekler, *Beginning JSP™, JSF™, and Tomcat Web Development: From Novice to Professional*, New York: Apress, 2007.
- [6] *HTML & CSS: design and build websites*, Indianapolis: John Wiley & Sons, Inc., 2014.
- [7] L. Sommerville; *Software Engineering, 9th*, pearson, 2011.
- [8] N. S. Williams, *Professional Java for Web Applications*, John Wiley & Sons, Incorporated, February 2014.
- [9] J. Duckeet, *JavaScript & JQuery internactive front-end web development*, Indianapolis: John Wiley & Sons, Inc, 2014.
- [10] S. Shah, *Maven for Eclipse*, Packt Publishing, August, 2014.

- [11] (2016, December 15). Spring Framework [Online]. Available:
https://en.wikipedia.org/wiki/Spring_Framework.
- [12] (2016, December 15). SOAP [Online]. Available:
<https://en.wikipedia.org/wiki/SOAP>.
- [13] (2016, December 15). REST [Online]. Available:
https://en.wikipedia.org/wiki/Representational_state_transfer.
- [14] (2016, December 15). Hibernate (framework) [Online]. Available:
[https://en.wikipedia.org/wiki/Hibernate_\(framework\)](https://en.wikipedia.org/wiki/Hibernate_(framework)).
- [15] (2016, December 15). jQuery [Online]. Available:
<https://en.wikipedia.org/wiki/JQuery>.
- [16] (2016, December 15). BPMN [Online]. Available:
https://en.wikipedia.org/wiki/Business_Process_Model_and_Notation.
- [17] (2016, December 15). Maven [Online]. Available:
<https://en.wikipedia.org/wiki/Maven>.
- [18] (2016, December 15). Model-view-controller [Online]. Available:
<https://en.wikipedia.org/wiki/Model-view-controller>.
- [19] (2016, December 15). Eclipse [Online]. Available:
<https://en.wikipedia.org/wiki/Eclipse>.
- [20] (2016, December 15). Apache Tomcat [Online]. Available:
https://en.wikipedia.org/wiki/Apache_Tomcat.
- [21] (2016, December 15). MySQL [Online]. Available:
<https://en.wikipedia.org/wiki/MySQL>.

[22] (2016, December 15). Unified Modeling Language [Online]. Available:
https://en.wikipedia.org/wiki/Unified_Modeling_Language.