A PROJECT REPORT ON

PMT Training and Internship

SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology

IN COMPUTER SCIENCE ENGINEERING

Submitted by

Yash Agarwal (1805314)

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
(Deemed to be University)

BHUBANESWAR
Nov 2021

KIIT UNIVERSITY

Certificate

This is to Certify that **Yash Agarwal (1805314)**, in partial fulfillment of the requirement for the degree of Bachelor of Technology in Computer Science and Engineering at the School of Computer Engineering, KIIT UNIVERSITY, Bhubaneswar during academic year 2021 has carried out projects entitled **PMT Training and Internship** and worked with the team of **PMT** for **HighRadius Technologies Private Limited**, Bhubaneswar during the Internship at the company and sincerely completed the assigned tasks.

Date **: 22-11-2021**

Acknowledgement

I would like to take this opportunity to thank all my sources of aspiration during the course of the internship.

I am grateful to **Mr. Anil Pradhan**, who gave me an opportunity to work on projects at HighRadius Technologies and for their continuous support during the internship and for their patience, motivation and immense knowledge. They helped us and guided us throughout the internship and development.

I hereby take the privilege to express my gratitude to all the people who were directly or indirectly involved in the execution of this work without whom this project would not have been a success.

I am also thankful to my senior developers and team leads for their valuable guidance, support, and cooperation extended by them. Then I would like to thank my project team members for their kind cooperation, help and never-ending support.

I am also thankful to KIIT Bhubaneswar for providing me with technical skills and facilities which proved to be very useful for our project.

Yash Agarwal
1805314

Abstract

The PMT team is responsible for creating and managing a Payment gateway for HighRadius wherein, different types of processors are used to communicate between the banks and provide a smooth experience.

There is an UI for the admins and also a UI for the users to achieve the payment. The entire UI is mainly created in ExtJS and the backend is based on JAVA and other technologies. Different technologies like Struts, Spring, Hibernate, etc are used to achieve the functionality and also follow a certain design pattern for faster and efficient development.

The HighRadius Payment gateway supports a huge number of payment methods. We were given technical as well as theoretical knowledge about the same in the process of training and internship.

Contents

Chapter 1

Introduction

1.1. Company Overview

HighRadius is a Fintech enterprise Software-as-a-Service (SaaS) company that provides an Integrated Receivables Platform to optimize receivables and payments functions such as credit, collections, cash application, deductions, and electronic billing and payment processing. The Integrated Receivables platform allows suppliers to digitally connect with buyers via the RadiusOne network, closing the loop from supplier receivable processes to buyer payable processes. HighRadius solutions have a proven track record of reducing Days Sales Outstanding (DSO) and bad debt, and increasing the operation efficiency, enabling companies to achieve an ROI in just a few months.

The goal is to help A/R and credit departments adopt innovative processes supported by high levels of automation so they may become more strategic, more streamlined, and more successful. It operates on three core principles: to reduce the total cost of ownership (TCO) of receivables solutions, to deliver a concrete return on investment (ROI) and fast payback periods to our customers, and to provide innovative functionality to the market. HighRadius is trusted by some of the world's largest corporations and is consistently named one of the fastest growing technology companies in Houston, where it is headquartered.

HighRadius offers two product lines as well as implementation services. Receivables Cloud and Payments Cloud are SaaS-based solution suites that automate and improve cash application, invoicing, and credit, collections, and deductions management. HighRadius Accelerators are certified solutions for SAP that enhance the automation available in the Receivables Management modules and reside natively in the application. HighRadius provides the most highly specialized expertise in SAP Receivables Management (formerly FSCM) and offers services to implement or optimize each of its modules.

Inaccuracy in the cash application process can lead to slowdowns and problems throughout the entire receivables lifecycle. Customer satisfaction is enhanced by more accuracy in the way invoices, collections, and deductions are handled. These improvements all contribute to the exceptional return on investment that our customers experience and the very high level of repeat business they give us.

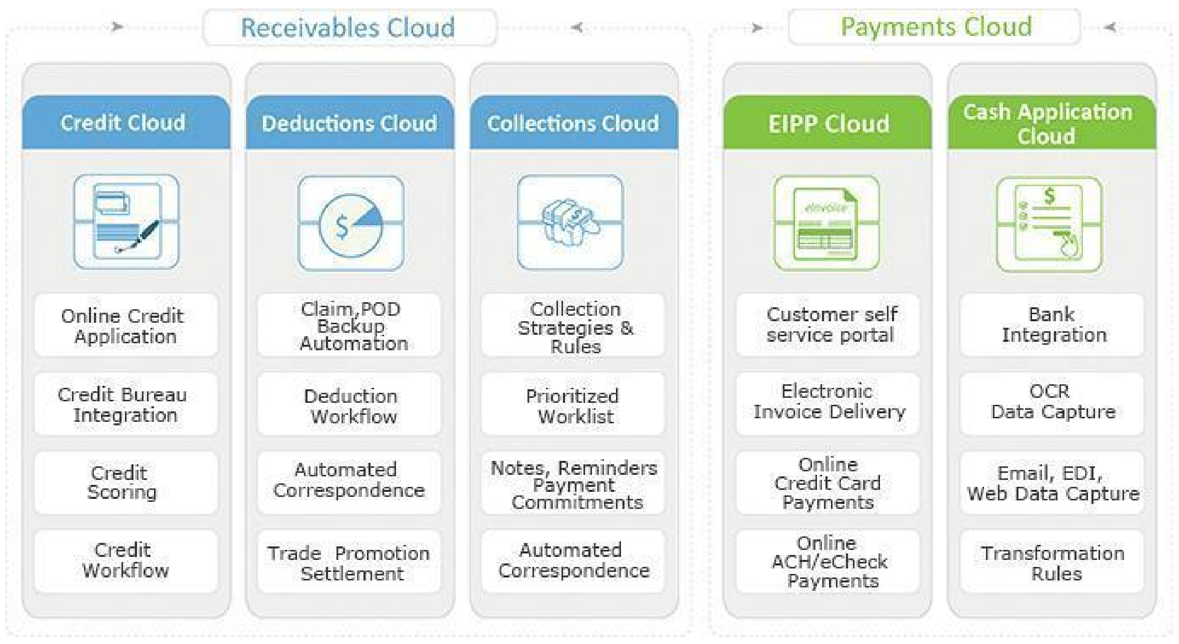

Receivables Cloud
Payments Cloud
Credit Cloud
Deductions Cloud
Collections Cloud
EIPP Cloud
Cash Application Cloud
Online Credit Application
Credit Bureau Integration
Credit Scoring
Credit Workflow
Claim,POD Backup Automation
Deduction Workflow
Automated Correspondence
Trade Promotion Settlement
Collection Strategies & Rules
Prioritized Worklist
Notes, Reminders Payment Commitments
Automated Correspondence
Customer self service portal
Electronic Invoice Delivery
Online Credit Card Payments
Online ACH/eCheck Payments
Bank Integration
OCR Data Capture
Email, EDI, Web Data Capture
Transformation Rules

Figure 1 | HighRadius solutions in a nutshell

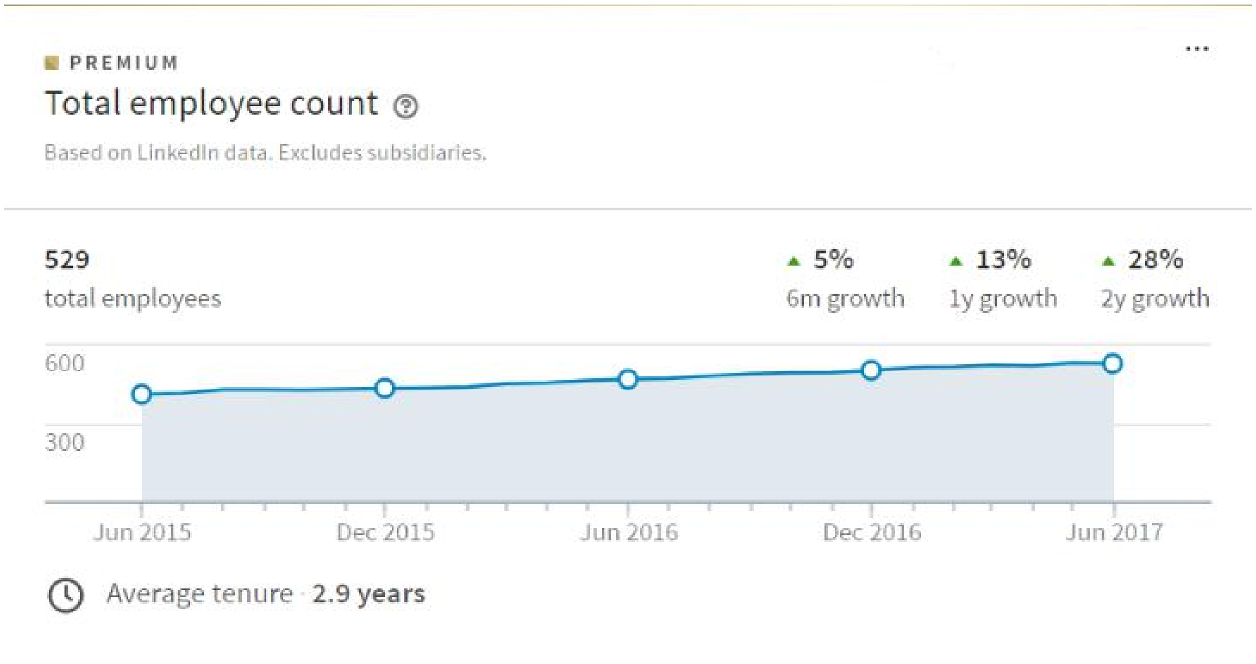
1.2. Career at HighRadius

HighRadius is an early-stage company providing software solutions for automating a business' order-to-cash cycle. For most businesses, accounts receivable is either the largest or the second largest asset on their balance sheet (in fact, it accounts for 40%, on average, of all the assets by value). Efficient management of accounts receivable has a direct impact on the financial health of a business. HighRadius is dedicated to helping businesses efficiently manage this asset.

The products uniquely complement traditional ERP systems and are delivered both as a service over the web, deployed as software-as-a-service (SaaS) in the cloud, and as add-ons to standard ERP functionality, deployed on-premises in the business' ERP landscape. Over the last few years, these innovative products have gained significant traction in the market and the company is now in an accelerated growth phase.

HighRadius seeks great minds, across different functions, to be a part of the growth story. At HighRadius, you get:

- an opportunity to build innovative products that customers love
- a challenging work environment


PREMIUM

Total employee count

Based on LinkedIn data. Excludes subsidiaries.

529 total employees

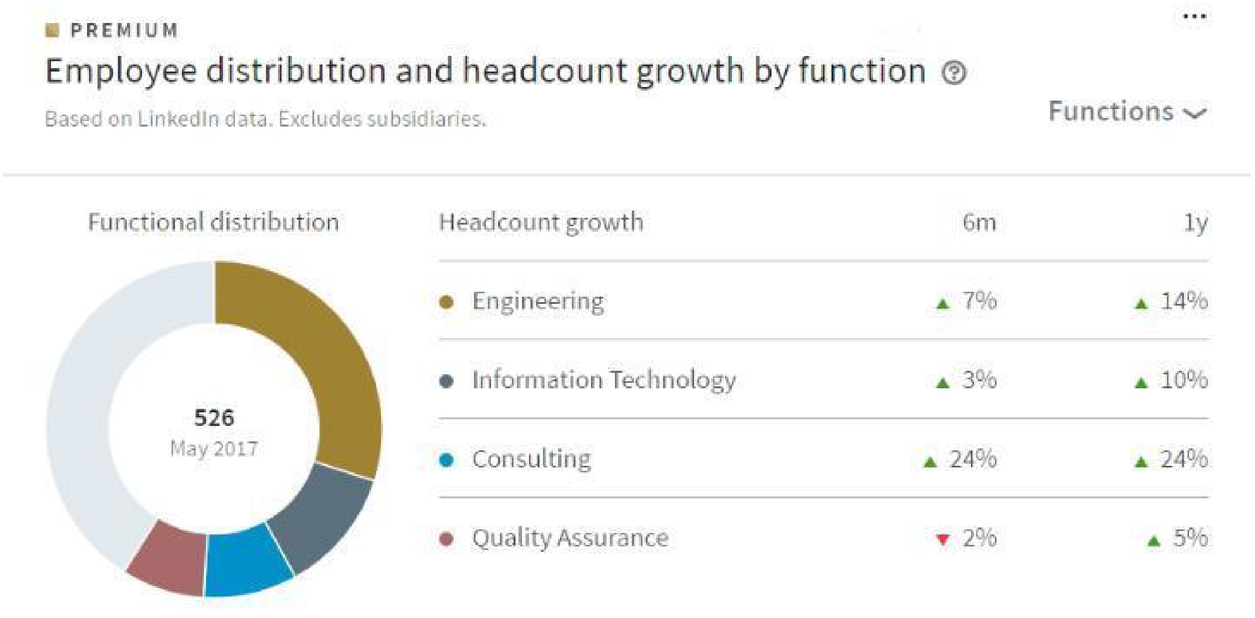
▲ 5% 6m growth

▲ 13% 1y growth

▲ 28% 2y growth

Average tenure · 2.9 years

Figure 2 | Total employee count


PREMIUM

Employee distribution and headcount growth by function

Based on LinkedIn data. Excludes subsidiaries.

Headcount growth	6m	1y
● Engineering	▲ 7%	▲ 14%
● Information Technology	▲ 3%	▲ 10%
● Consulting	▲ 24%	▲ 24%
● Quality Assurance	▼ 2%	▲ 5%

Figure 3 | Employee distribution and headcount growth by function

Chapter 2

Struts2

Apache Struts 2 is an elegant, extensible framework for creating enterprise-ready Java web applications. This framework is designed to streamline the full development cycle from building, to deploying and maintaining applications over time. Apache Struts 2 was originally known as Web Work 2.

Struts2 is a popular and mature web application framework based on the MVC design pattern. Struts2 is not just a new version of Struts 1, but it is a complete rewrite of the Struts architecture.

The Webwork framework initially started with Struts framework as the basis and its goal was to offer an enhanced and improved framework built on Struts to make web development easier for the developers.

After a while, the Webwork framework and the Struts community joined hands to create the famous Struts2 framework.

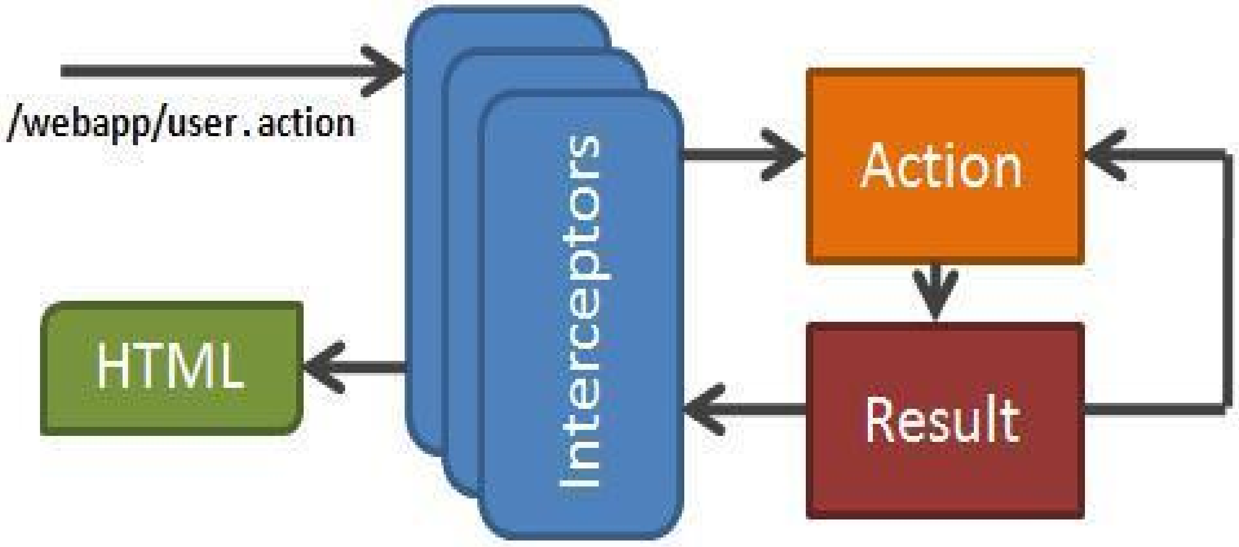


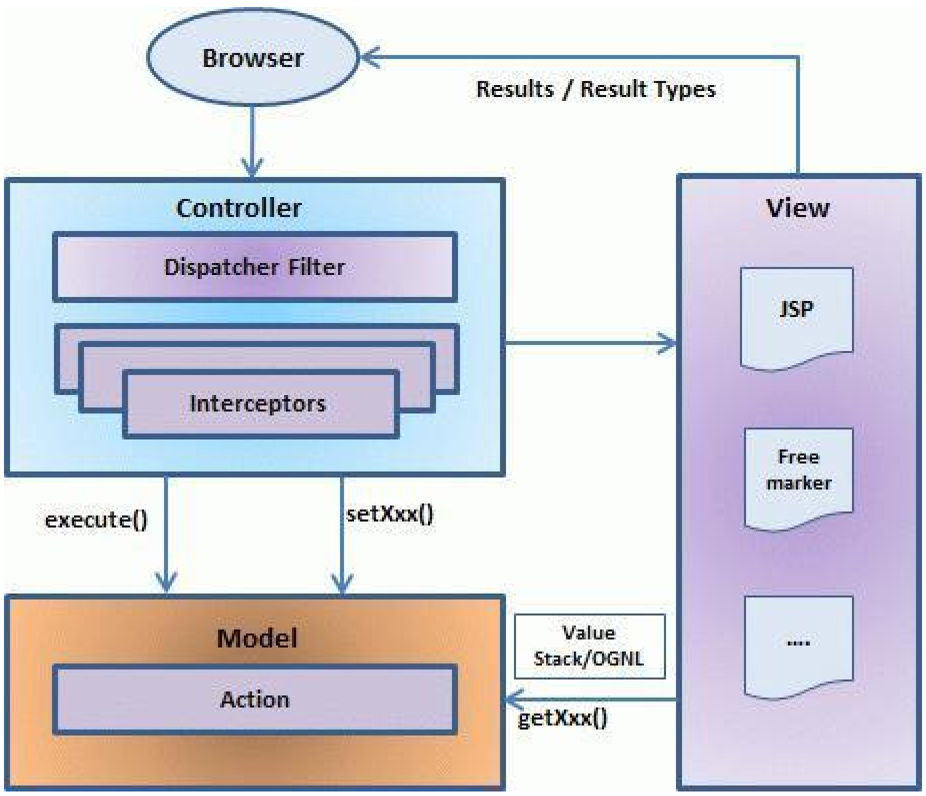
Figure 4 | Struts-request-cyclel

2.1. Struts Architecture

From a high level, Struts2 is a pull-MVC (or MVC2) framework. The Model-ViewController pattern in Struts2 is implemented with the following five core components −

- Actions
- Interceptors
- Value Stack / OGNL
- Results / Result types
- View technologies

Struts 2 is slightly different from a traditional MVC framework, where the action takes the role of the model rather than the controller, although there is some overlap.



The above diagram depicts the Model, View and Controller to the Struts2 high level architecture. The controller is implemented with a Struts2 dispatcher servlet filter as well as interceptors, this model is implemented with actions, and the view is a combination of result types and results. The value stack and OGNL provides common thread, linking and enabling integration between the other components.

Apart from the above components, there will be a lot of information that relates to configuration. Configuration for the web application, as well as configuration for actions, interceptors, results, etc.

This is the architectural overview of the Struts 2 MVC pattern. We will go through each component in more detail in the subsequent chapters.

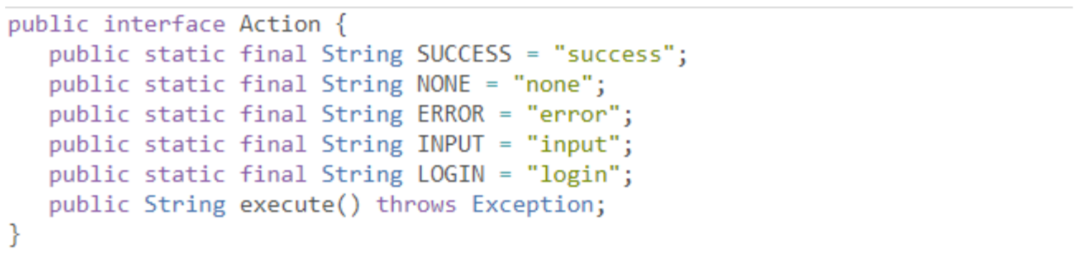
2.2. Action in Struts

Actions are the core of the Struts2 framework, as they are for any MVC (Model View Controller) framework. Each URL is mapped to a specific action, which provides the processing logic which is necessary to service the request from the user.

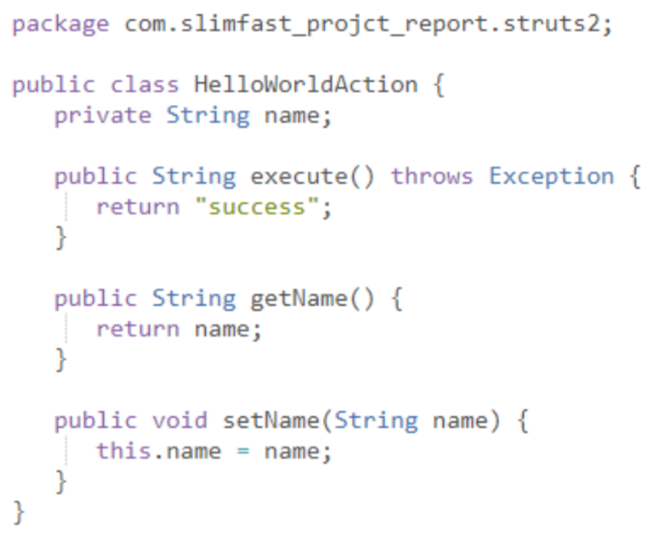
But the action also serves in two other important capacities. Firstly, the action plays an important role in the transfer of data from the request through to the view, whether it's a JSP or other type of result. Secondly, the action must assist the framework in determining which result should render the view that will be returned in the response to the request.

The only requirement for actions in Struts2 is that there must be one no argument method that returns either a String or Result object and must be a POJO. If the no-argument method is not specified, the default behavior is to use the execute() method.

Optionally you can extend the ActionSupport class which implements six interfaces including Action interface. The Action interface is as follows −

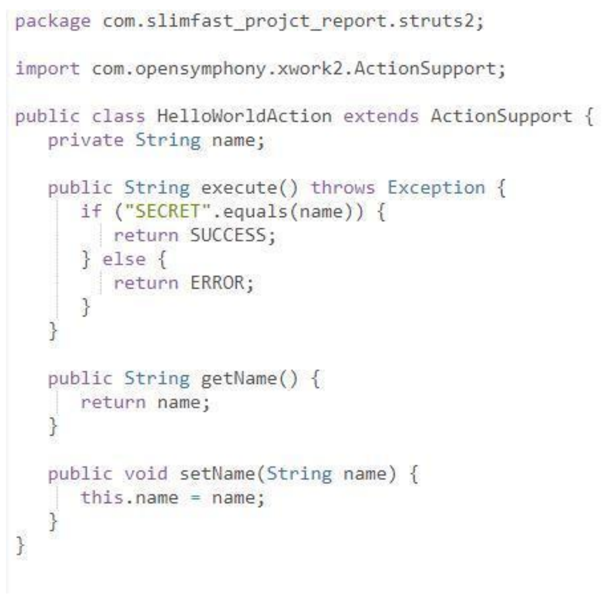


```java
public interface Action {
   public static final String SUCCESS = "success";
   public static final String NONE = "none";
   public static final String ERROR = "error";
   public static final String INPUT = "input";
   public static final String LOGIN = "login";
   public String execute() throws Exception;
}
```



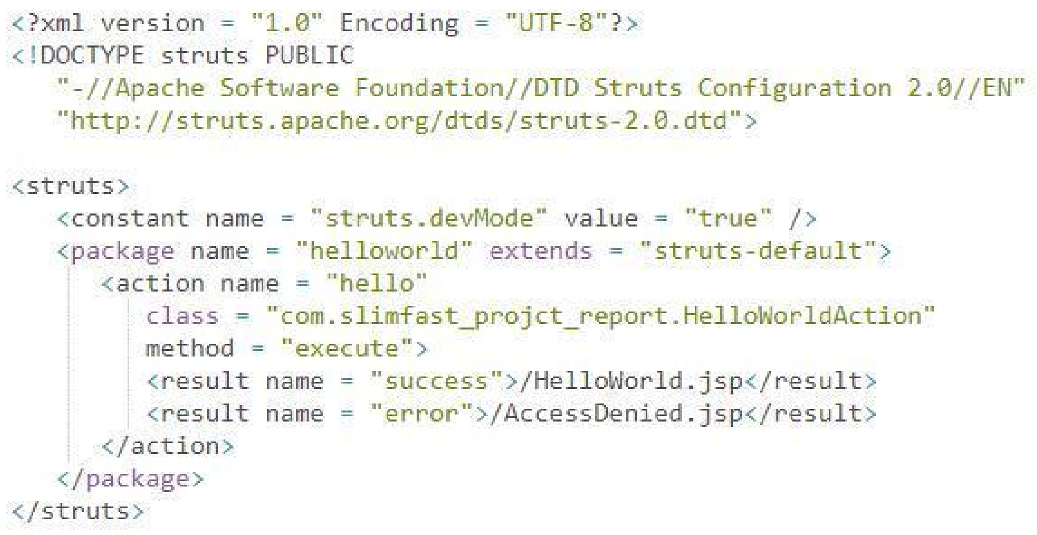
```java
package com.slimfast_projct_report.struts2;

public class HelloWorldAction {
   private String name;

   public String execute() throws Exception {
      return "success";
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }
}
```

To illustrate the point that the action method controls the view, let us make the following change to the execute method and extend the class ActionSupport as follows −



```java
package com.slimfast_projct_report.struts2;

import com.opensymphony.xwork2.ActionSupport;

public class HelloWorldAction extends ActionSupport {
   private String name;

   public String execute() throws Exception {
      if ("SECRET".equals(name)) {
         return SUCCESS;
      } else {
         return ERROR;
      }
   }

   public String getName() {
      return name;
   }

   public void setName(String name) {
      this.name = name;
   }
}
```

we have some logic in the execute method to look at the name attribute. If the attribute equals the string "SECRET", we return SUCCESS as the result otherwise we return ERROR as the result. Because we have extended ActionSupport, so we can use String constants SUCCESS and ERROR. struts.xml file as follows −



```
<?xml version = "1.0" Encoding = "UTF-8"?>
<!DOCTYPE struts PUBLIC
   "-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"
   "http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>
   <constant name = "struts.devMode" value = "true" />
   <package name = "helloworld" extends = "struts-default">
      <action name = "hello"
         class = "com.slimfast_projct_report.HelloWorldAction"
         method = "execute">
         <result name = "success">/HelloWorld.jsp</result>
         <result name = "error">/AccessDenied.jsp</result>
      </action>
   </package>
</struts>
```

2.3. Struts in HighRadius

Struts is a useful framework which was used to render action class using the url mentioned in the Javascript. The Struts were implemented in the company’s framework and were developed with Java technologies at the backend. The front end used to accommodate the ExtJS. ExtJS is a pure JavaScript application framework for building interactive cross-platform web applications using techniques such as Ajax, DHTML and DOM scripting.

Using Struts all the API are synced nicely to the structure and provide developers an easy way to handle and work along them.

Chapter 3

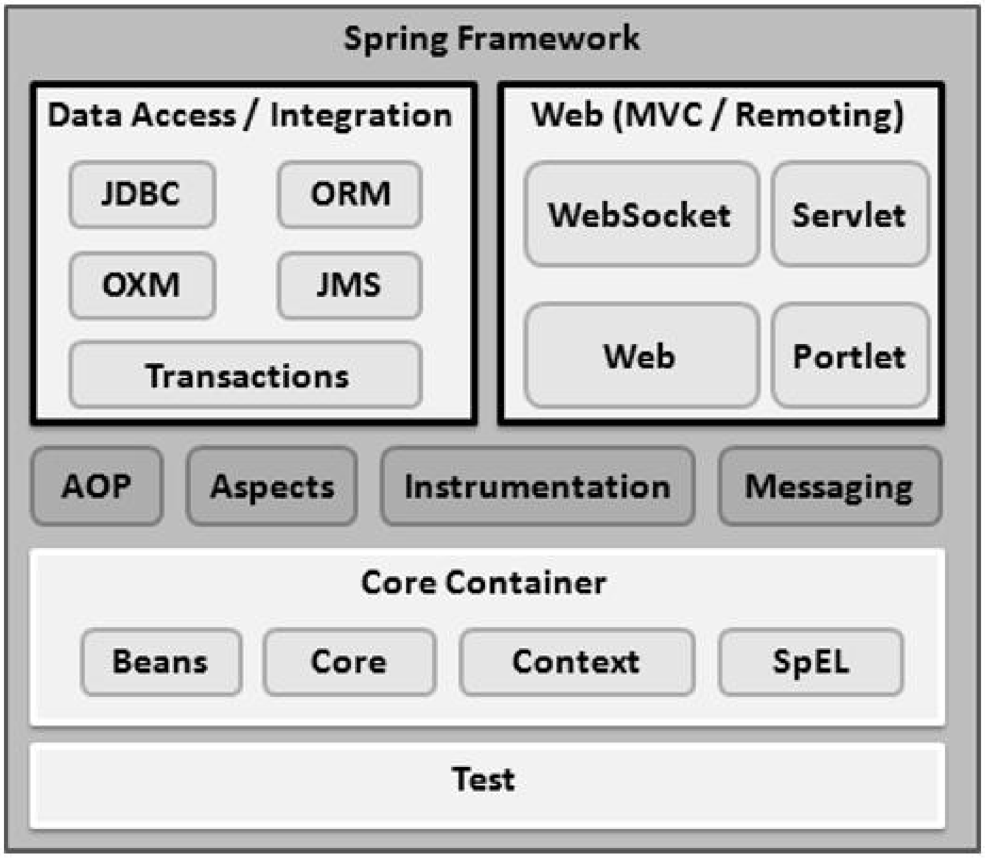
Spring

Spring framework is an open source Java platform that provides comprehensive infrastructure support for developing robust Java applications very easily and very rapidly. Spring framework was initially written by Rod Johnson and was first released under the Apache 2.0 license in June 2003.Spring is lightweight when it comes to size and transparency. The basic version of Spring framework is around 2MB.

The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework aims to make J2EE development easier to use and promotes good programming practices by enabling a POJO-based programming model.

3.1. Spring Architecture

Spring could potentially be a one-stop shop for all your enterprise applications. However, Spring is modular, allowing you to pick and choose which modules are applicable to you, without having to bring in the rest. The Spring Framework provides about 20 modules which can be used based on an application requirement.



3.2. Core Container

The Core Container consists of the Core, Beans, Context, and Expression Language modules the details of which are as follows −

- The Core module provides the fundamental parts of the framework, including the IoC and Dependency Injection features.
- The Bean module provides BeanFactory, which is a sophisticated implementation of the factory pattern.
- The Context module builds on the solid base provided by the Core and Beans modules and it is a medium to access any objects defined and configured. The ApplicationContext interface is the focal point of the Context module.

- The SpEL module provides a powerful expression language for querying and manipulating an object graph at runtime.

3.3. Data Access/Integration

The Data Access/Integration layer consists of the JDBC, ORM, OXM, JMS and Transaction modules whose detail is as follows −

- The JDBC module provides a JDBC-abstraction layer that removes the need for tedious JDBC related coding.
- The ORM module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, Hibernate, and iBatis.
- The OXM module provides an abstraction layer that supports Object/XML mapping implementations for JAXB, Castor, XMLBeans, JiBX and XStream.
- The Java Messaging Service JMS module contains features for producing and consuming messages.

- The Transaction module supports programmatic and declarative transaction management for classes that implement special interfaces and for all your POJOs.

3.4. Web

The Web layer consists of the Web, Web-MVC, Web-Socket, and Web-Portlet modules the details of which are as follows −

- The Web module provides basic web-oriented integration features such as multipart file-upload functionality and the initialization of the IoC container using servlet listeners and a web-oriented application context.
- The Web-MVC module contains Spring's Model-View-Controller (MVC) implementation for web applications.
- The Web-Socket module provides support for WebSocket-based, two-way communication between the client and the server in web applications.

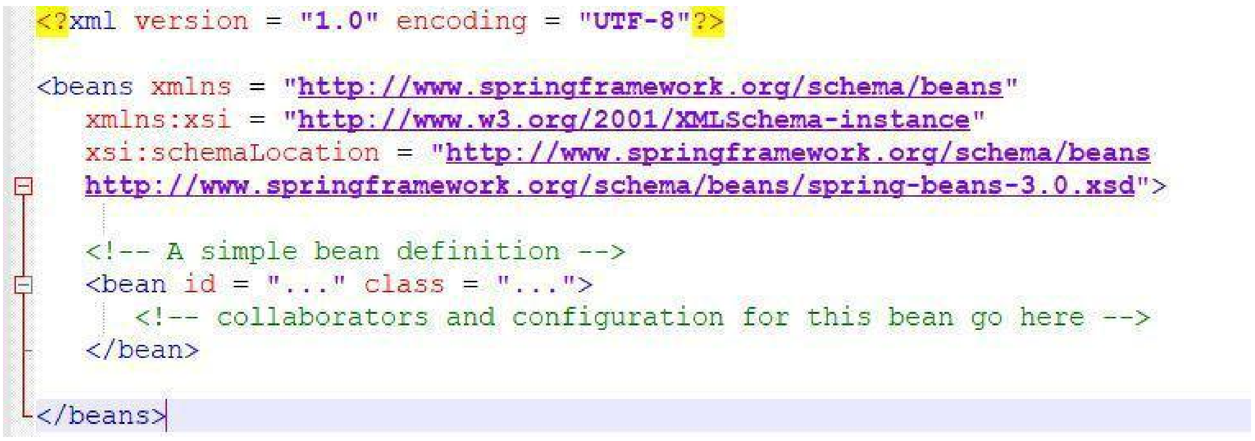
- The Web-Portlet module provides the MVC implementation to be used in a portlet environment and mirrors the functionality of the Web-Servlet module.

3.5. Spring - Bean Definition

The objects that form the backbone of an application and that are managed by the Spring IoC container are called beans. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container. These beans are created with the configuration metadata that you supply to the container. Bean definition contains the information called configuration metadata.

Spring IoC container is totally decoupled from the format in which this configuration metadata is actually written. Following are the three important methods to provide configuration metadata to the Spring Container −

- XML based configuration file.
- Annotation-based configuration
- Java-based 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"
   xsi:schemaLocation = "http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

   <!-- A simple bean definition -->
   <bean id = "..." class = "...">
      <!-- collaborators and configuration for this bean go here -->
   </bean>

</beans>
```

3.6 Spring- JDBC

While working with the database using plain old JDBC, it becomes cumbersome to write unnecessary code to handle exceptions, opening and closing database connections, etc. However, Spring JDBC Framework takes care of all the low-level details starting from opening the connection, preparing and executing the SQL statement, process exceptions, handling transactions and finally closing the connection.

JdbcTemplate Class

The JDBC Template class executes SQL queries, updates statements, stores procedure calls, performs iteration over ResultSets, and extracts returned parameter values. It also catches JDBC exceptions and translates them to the generic, more informative, exception hierarchy defined in the org.springframework.dao package.

Instances of the *JdbcTemplate* class are *thread-safe* once configured. So one can configure a single instance of a *JdbcTemplate* and then safely inject this shared reference into multiple DAOs.

A DataSource supply is needed to the JDBC Template so it can configure itself to get database access. One can configure the DataSource in the XML file with a piece of code as shown in the following code snippet −

```
<bean id = "dataSource"
   class = "org.springframework.jdbc.datasource.DriverManagerDataSource">
   <property name = "driverClassName" value = "com.mysql.jdbc.Driver"/>
   <property name = "url" value = "jdbc:mysql://localhost:3306/TEST"/>
   <property name = "username" value = "root"/>
   <property name = "password" value = "password"/>
</bean>
```

Data Access Object (DAO)

DAO stands for Data Access Object, which is commonly used for database interaction. DAOs exist to provide a means to read and write data to the database and they should expose this functionality through an interface by which the rest of the application will access them.

The DAO support in Spring makes it easy to work with data access technologies like JDBC, Hibernate, JPA, or JDO in a consistent way.

Executing SQL statements

The different SQL CRUD Operations that can be executed using this framework are as follows:

Querying for an integer

```
String SQL = "select count(*) from Student";
```

```
int rowCount = jdbcTemplateObject.queryForInt( SQL );
```

Querying for a String

```
String SQL = "select name from Student where id = ?";

String name = jdbcTemplateObject.queryForObject(SQL, new Object[]{10},
String.class);
```

Querying and returning an object

```
String SQL = "select * from Student where id = ?";
Student student = jdbcTemplateObject.queryForObject(

  SQL, new Object[]{10}, new StudentMapper());

public class StudentMapper implements RowMapper<Student> {

  public Student mapRow(ResultSet rs, int rowNum) throws SQLException
    { Student student = new Student();

    student.setID(rs.getInt("id"));

    student.setName(rs.getString("name"));

    student.setAge(rs.getInt("age"));

    return student;

  }

}
```

Inserting a row into the table

```java
String SQL = "insert into Student (name, age) values (?, ?)";
jdbcTemplateObject.update( SQL, new Object[]{"Zara", 11} );
```

Updating a row into the table

```java
String SQL = "update Student set name = ? where id = ?";
jdbcTemplateObject.update( SQL, new Object[]{"Zara", 10} );
```

Deleting a row from the table

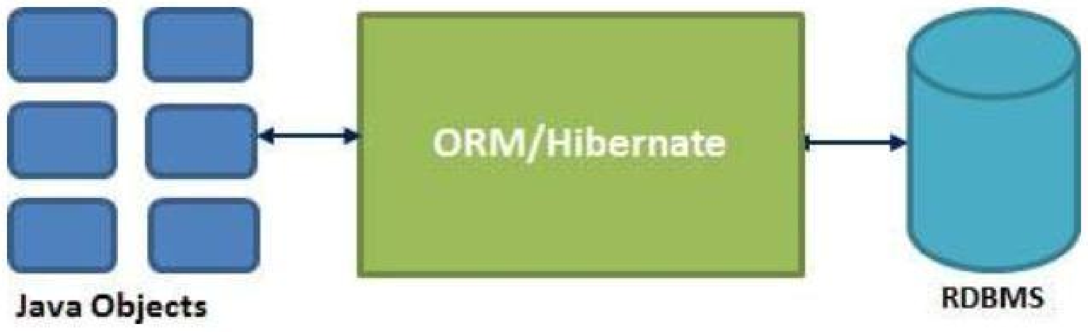
```java
String SQL = "delete Student where id = ?";
jdbcTemplateObject.update( SQL, new Object[]{20} );
```

Chapter 4

Hibernate

Hibernate is an Object-Relational Mapping (ORM) solution for JAVA. It is an open source persistent framework created by Gavin King in 2001. It is a powerful, high performance Object-Relational Persistence and Query service for any Java Application.
It maps Java classes to database tables and from Java data types to SQL data types and relieves the developer from 95% of common data persistence related programming tasks.

It sits between traditional Java objects and database server to handle all the works in persisting those objects based on the appropriate O/R mechanisms and patterns.

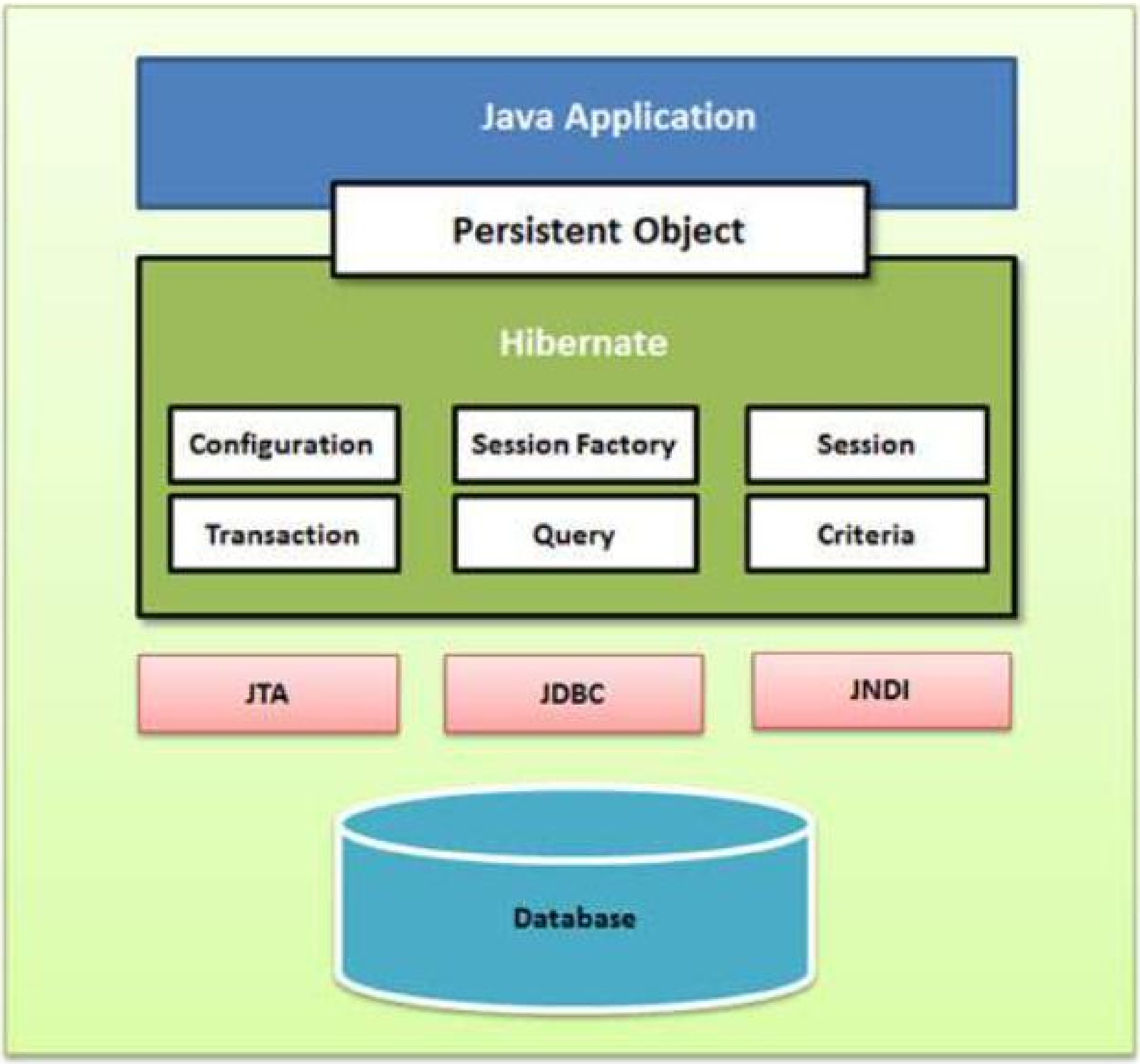


Advantages of Hibernate:

- Hibernate takes care of mapping Java classes to database tables using XML files and without writing any line of code.

- Provides simple APIs for storing and retrieving Java objects directly to and from the database.
- If there is change in the database or in any table, then you need to change the XML file properties only.
- Abstracts away the unfamiliar SQL types and provides a way to work around familiar Java Objects.

- Hibernate does not require an application server to operate.
- Manipulates Complex associations of objects of your database.
- Minimizes database access with smart fetching strategies.
- Provides simple querying of data.



Hibernate has a layered architecture which helps the user to operate without having to know the underlying APIs. Hibernate makes use of the database and configuration data to provide persistence services (and persistent objects) to the application.

4.1 Class objects involved in Hibernate Application Architecture

Configuration Object:

The Configuration object is the first Hibernate object we create in any Hibernate application. It is usually created only once during application initialization. It represents a configuration or properties file required by Hibernate. The Configuration object provides two keys components −

Database Connection − This is handled through one or more configuration files supported by Hibernate. These files are hibernate.properties and hibernate.cfg.xml. Class Mapping Setup − This component creates the connection between the Java classes and database tables.

SessionFactory Object:

Configuration object is used to create a SessionFactory object which in turn configures Hibernate for the application using the supplied configuration file and allows for a Session object to be instantiated. The SessionFactory is a thread safe object and used by all the threads of an application.

The SessionFactory is a heavyweight object; it is usually created during application start up and kept for later use. You would need one SessionFactory object per database using a separate configuration file. So, if you are using multiple databases, then you would have to create multiple SessionFactory objects.

Session Object:

A Session is used to get a physical connection with a database. The Session object is lightweight and designed to be instantiated each time an interaction is needed with the database. Persistent objects are saved and retrieved through a Session object.

The session objects should not be kept open for a long time because they are not usually thread safe and they should be created and destroyed as needed.

Transaction Object:

A Transaction represents a unit of work with the database and most of the RDBMS supports transaction functionality. Transactions in Hibernate are handled by an underlying transaction manager and transaction (from JDBC or JTA).

This is an optional object and Hibernate applications may choose not to use this interface, instead managing transactions in their own application code.

Query Object:

Query objects use SQL or Hibernate Query Language (HQL) string to retrieve data from the database and create objects. A Query instance is used to bind query parameters, limit the number of results returned by the query, and finally to execute the query.

Criteria Object:

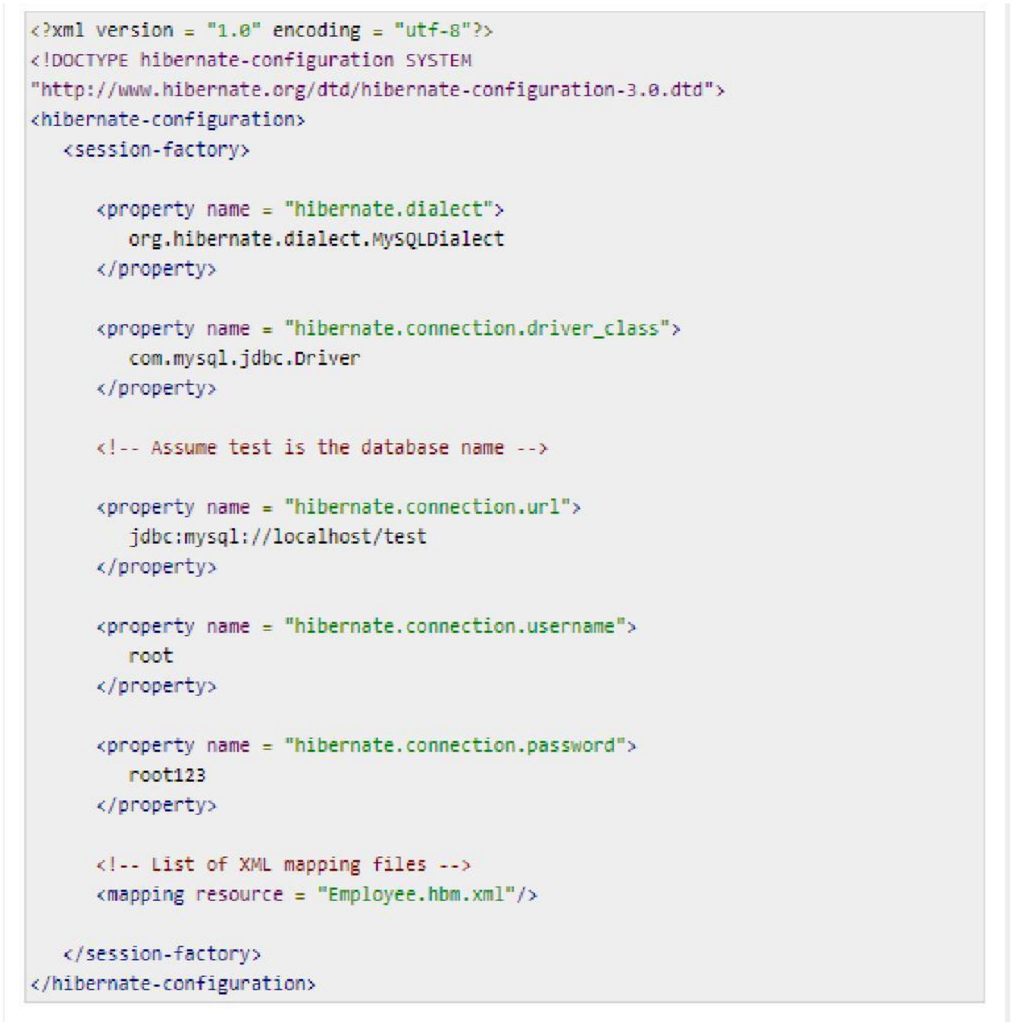
Criteria objects are used to create and execute object oriented criteria queries to retrieve objects.

4.2 Hibernate in Highradius:

Hibernate requires knowing in advance — where to find the mapping information that defines how your Java classes relate to the database tables. Hibernate also requires a set of configuration settings related to database and other related parameters. All such information is usually supplied as a standard Java properties file called hibernate.properties, or as an XML file named hibernate.cfg.xml.

We considered XML formatted file hibernate.cfg.xml to specify required Hibernate properties in our examples.

4.3 Hibernate with MySQL Database:



```xml
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE hibernate-configuration SYSTEM
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
   <session-factory>

      <property name = "hibernate.dialect">
         org.hibernate.dialect.MySQLDialect
      </property>

      <property name = "hibernate.connection.driver_class">
         com.mysql.jdbc.Driver
      </property>

      <!-- Assume test is the database name -->

      <property name = "hibernate.connection.url">
         jdbc:mysql://localhost/test
      </property>

      <property name = "hibernate.connection.username">
         root
      </property>

      <property name = "hibernate.connection.password">
         root123
      </property>

      <!-- List of XML mapping files -->
      <mapping resource = "Employee.hbm.xml"/>

   </session-factory>
</hibernate-configuration>
```

Chapter 5

User Stories

5.1 PROBLEM STATEMENT:

US 1: Developing a full stack movie management application
US 2: Developing a full stack Invoice management application using RRDMS
US 3: POC for a Payment processor in Argentina and Chile locations
US 4: Integration of a new processor into the HighRadius payment gateway

5.2 Technologies Used:

1. ExtJS
2. Struts
3. Hibernate
4. Spring
5. MySQL
6. XML
7. SOAP UI/ Postman

For client-side, our corporation uses **ExtJS** Framework. ExtJS is a javascript framework (client-side) that enables developers to develop Rich Internet Applications (RIA) (static websites or data-driven applications) with a large number of options. ExtJS has a huge collection of controls (ranging from textboxes to highly sophisticated UI Controls).

For data parsing between client-side and server-side, **JSON** (JavaScript Object Notation) is used. JSON is the lightweight data-interchange format used in parsing data between different platforms. It is easy to parse and generate and better than XML. JSON is a text format that is completely language independent. JSON is also native to JavaScript.

For server-side, our corporation uses **Java**. Java is the most widely used programming language for applications in the world.It has APIs for different areas. Since it is an Object Oriented programming language, all these features are also very beneficial in developing a large application.

5.3 Client-Side (ExtJS Framework)

ExtJS is a JavaScript framework (client-side) that enables developers to develop Rich Internet Applications (RIA) (static websites or data-driven applications) with a large number of options. Since ExtJS is a javascript framework, all of the javascript rules are applicable for ExtJS. ExtJS makes excellent & extensive use on DOM, CSS etc.

Ext JS includes a set of GUI-based form controls (or "widgets") for use within web applications:

- text field and text area input controls

- date fields with a pop-up date picker
- numeric fields
- list box and combo boxes
- radio and checkbox controls
- HTML editor control
- grid control (with both read-only and edit modes, sortable data, lockable and draggable columns, and a variety of other features)

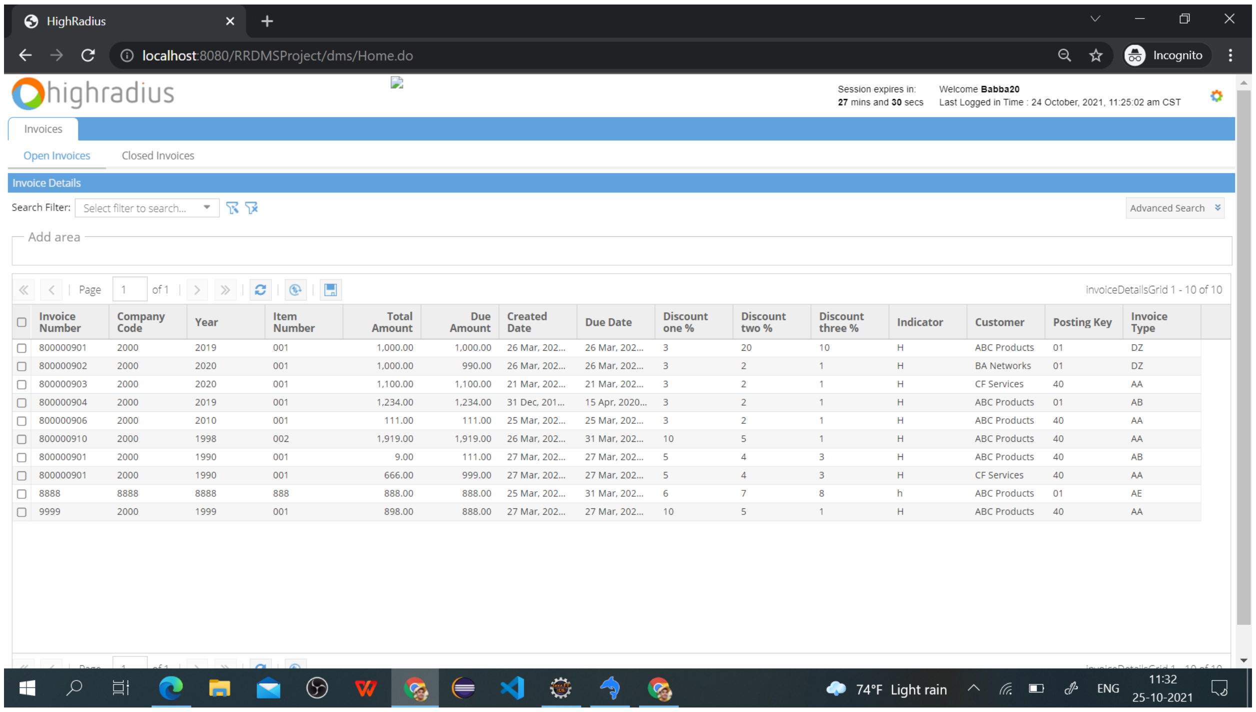
- tree control
- tab panels
- toolbars
- desktop application-style menus
- region panels to allow a form to be divided into multiple sub-sections
- Sliders
- vector graphics charts

Many of these controls can communicate with a web server using Ajax.

5.4 Server-side (Java)

Server side coding is covered through Java (which further comprises Hibernate, spring and struts framework). For all database operations, Hibernate is used through HQL (Hibernate Query Language) for fetching the data from the database. Spring is used to set the property of the objects and get the values from the UI using getter setters. Spring injects the data in the object taken as the property of Action class which indirectly calls manager operations through manager objects.

A Few Glimpses of my work



localhost:8080/RRDMSProject/dms/Home.do

Session expires in: 27 mins and 30 secs
Welcome Babba20
Last Logged in Time : 24 October, 2021, 11:25:02 am CST

Invoices — Open Invoices | Closed Invoices

Invoice Details

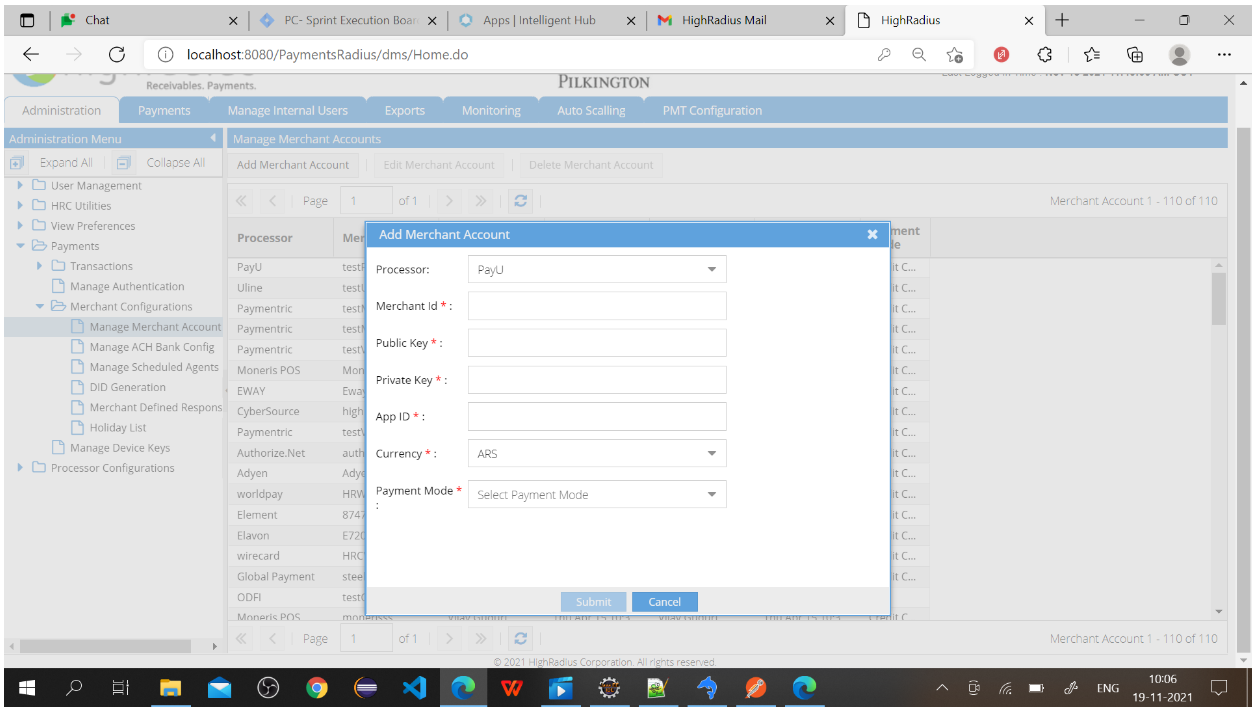
Search Filter: Select filter to search... | Advanced Search

Add area

Page 1 of 1 — invoiceDetailsGrid 1 - 10 of 10

Invoice Number	Company Code	Year	Item Number	Total Amount	Due Amount	Created Date	Due Date	Discount one %	Discount two %	Discount three %	Indicator	Customer	Posting Key	Invoice Type
800000901	2000	2019	001	1,000.00	1,000.00	26 Mar, 202...	26 Mar, 202...	3	20	10	H	ABC Products	01	DZ
800000902	2000	2020	001	1,000.00	990.00	26 Mar, 202...	26 Mar, 202...	3	2	1	H	BA Networks	01	DZ
800000903	2000	2020	001	1,100.00	1,100.00	21 Mar, 202...	21 Mar, 202...	3	2	1	H	CF Services	40	AA
800000904	2000	2019	001	1,234.00	1,234.00	31 Dec, 201...	15 Apr, 2020...	3	2	1	H	ABC Products	01	AB
800000906	2000	2010	001	111.00	111.00	25 Mar, 202...	25 Mar, 202...	3	2	1	H	ABC Products	40	AA
800000910	2000	1998	002	1,919.00	1,919.00	26 Mar, 202...	31 Mar, 202...	10	5	1	H	ABC Products	40	AA
800000901	2000	1990	001	9.00	111.00	27 Mar, 202...	27 Mar, 202...	5	4	3	H	ABC Products	40	AB
800000901	2000	1990	001	666.00	999.00	27 Mar, 202...	27 Mar, 202...	5	4	3	H	CF Services	40	AA
8888	8888	8888	888	888.00	888.00	25 Mar, 202...	31 Mar, 202...	6	7	8	h	ABC Products	01	AE
9999	2000	1999	001	898.00	888.00	27 Mar, 202...	27 Mar, 202...	10	5	1	H	ABC Products	40	AA

74°F Light rain — ENG 11:32 25-10-2021



localhost:8080/PaymentsRadius/dms/Home.do

Administration | Payments | Manage Internal Users | Exports | Monitoring | Auto Scalling | PMT Configuration

Administration Menu: User Management; HRC Utilities; View Preferences; Payments — Transactions; Manage Authentication; Merchant Configurations — Manage Merchant Account; Manage ACH Bank Config; Manage Scheduled Agents; DID Generation; Merchant Defined Respons; Holiday List; Manage Device Keys; Processor Configurations

Manage Merchant Accounts — Add Merchant Account | Edit Merchant Account | Delete Merchant Account — Merchant Account 1 - 110 of 110

Processor: PayU, Uline, Paymentric, Paymentric, Paymentric, Moneris POS, EWAY, CyberSource, Paymentric, Authorize.Net, Adyen, worldpay, Element, Elavon, wirecard, Global Payment, ODFI, Moneris POS

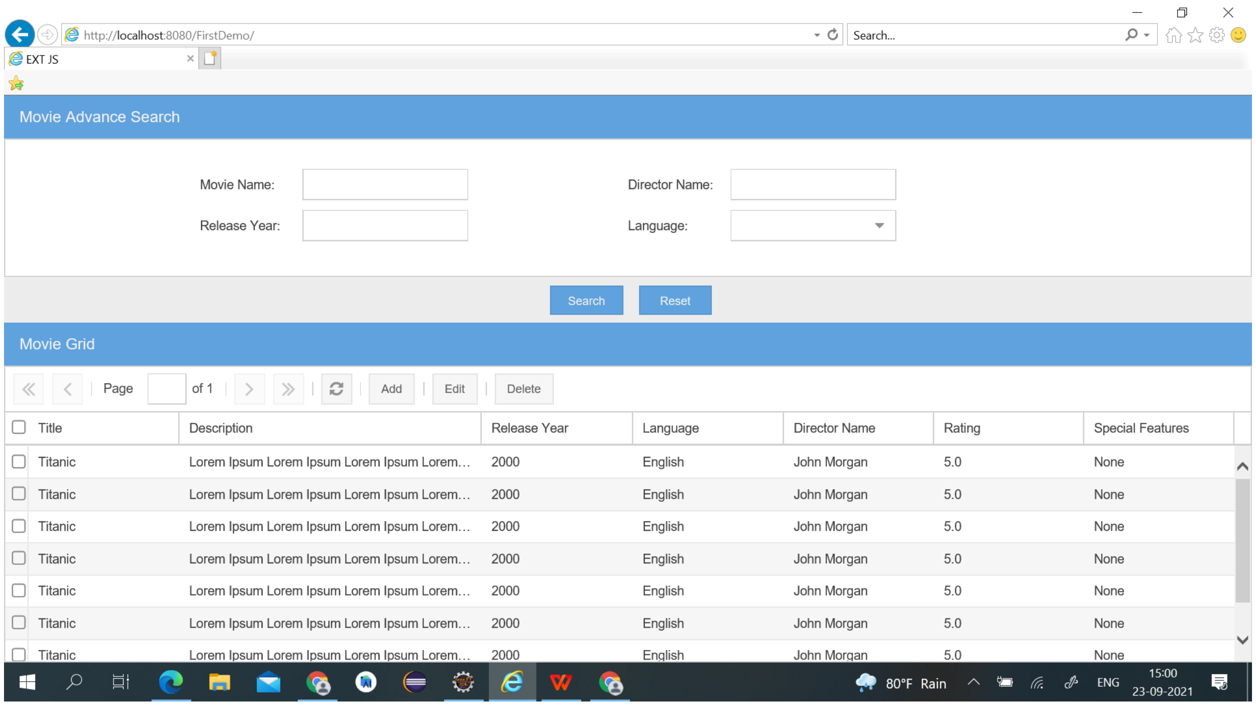
Add Merchant Account

Processor:	PayU
Merchant Id * :	
Public Key * :	
Private Key * :	
App ID * :	
Currency * :	ARS
Payment Mode * :	Select Payment Mode

Submit | Cancel

© 2021 HighRadius Corporation. All rights reserved.

ENG 10:06 19-11-2021



Movie Advance Search

Movie Name:		Director Name:	
Release Year:		Language:	

Movie Grid

Title	Description	Release Year	Language	Director Name	Rating	Special Features
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None
Titanic	Lorem Ipsum Lorem Ipsum Lorem Ipsum Lorem...	2000	English	John Morgan	5.0	None

Chapter 6

Conclusion

During the tenure of my internship, I had the exposure to a variety of work being done at HighRadius. HighRadius has really been a great journey for me for the past 10 months and I have got to learn a lot. I have also got to work with three different teams which has also helped me to get into tougher challenges.

I have got to know a lot about Web development and an overall full stack development in actual. I have got to work on ReactJS, Java, the calling infrastructure team and now the Payments team. I have also worked on a lot more things which may be confidential to the company so I have refrained from sharing those. Thankyou for this opportunity.