**What this project does??**

This project include almost all concepts in servlets and JSP technology viz. MVC design pattern, servlet as controller, filters, listeners, model, error handling using JSP, database handling using hibernate, logging mechanism etc. **This is pure xml based project** unlike project 'ServletJSPJDBCFullApp' **(Please note that we have used web.xml file in 'ServletJSPJDBCFullApp' application only for initializing context parameters as that is not at all possible with annotations.)**

This is replica of project 'ServletJSPJDBCFullApp' except it uses

1. xml based mapping

2. Hibernate instead of JDBC

**\*\*So in this project guide we have just explained only the pointes which are different than that of project 'ServletJSPJDBCFullApp'. For common part you have to refer project guide file of the project 'ServletJSPJDBCFullApp'.**

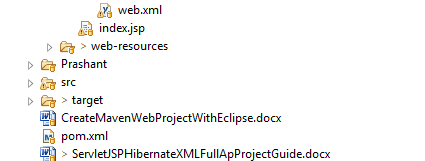
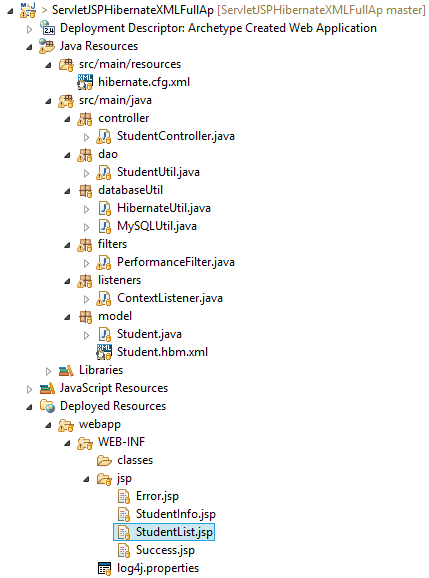
This project guide is divided into two sections.

In first section we have explained structural difference between this project and that of 'ServletJSPJDBCFullApp'.

In second section we have just listed the points covered in this project which are common with 'ServletJSPJDBCFullApp' project so please refer respective part of file 'ServletJSPJDBCFullAppProjectGuide.docx' in project 'ServletJSPJDBCFullApp'.

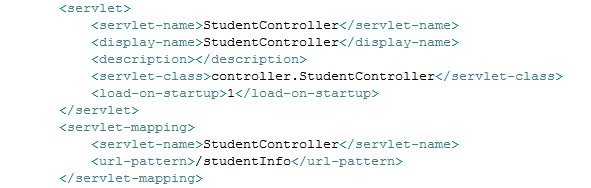
**Section I**

Project structure is as follows

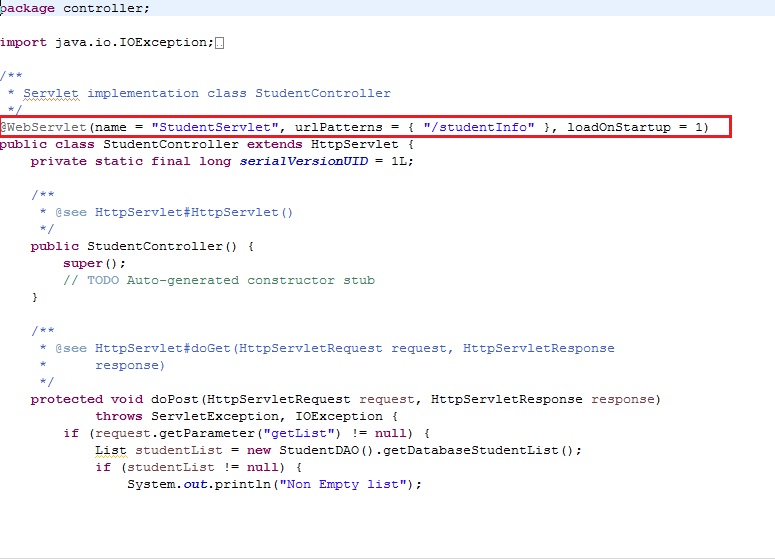


**1. Difference between web.xml and annotation based configuration of Servlet and JSP app like 'ServletJSPJDBCFullApp'.**

i) servlet mapping with web.xml



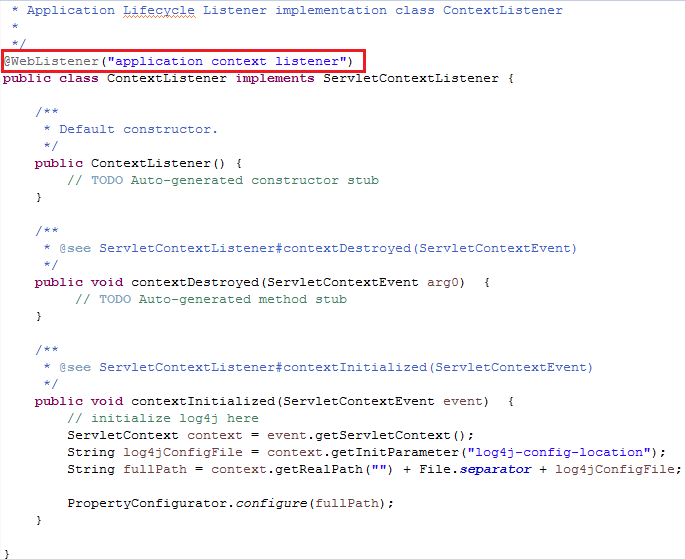
Annotation based mapping of servlet from project **'ServletJSPJDBCFullApp'**



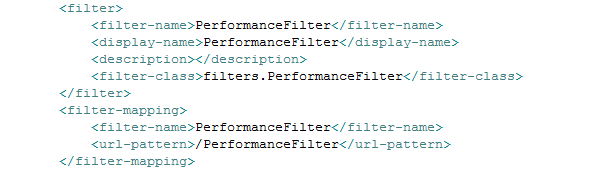
ii) Context Listener with web.xml



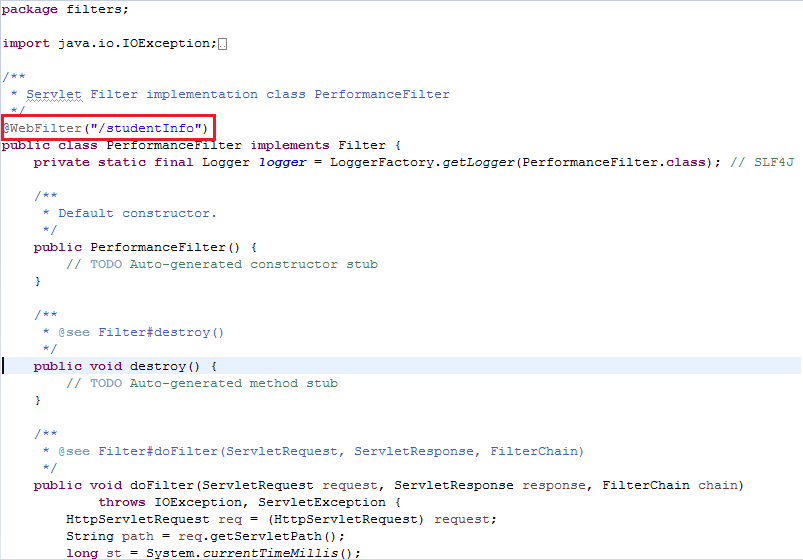
Annotation based mapping of context listener from project **'ServletJSPJDBCFullApp'**



iii) Filter with web.xml in this project



Annotation based mapping of servlet filter from project **'ServletJSPJDBCFullApp'**

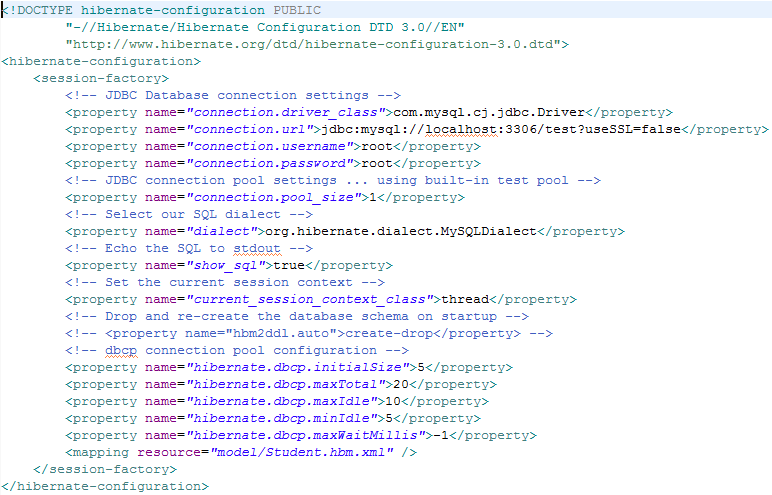


iv) Context parameter declaration : This is only the common part in web.xml file of this project and that of project ' ServletJSPJDBCFullApp'

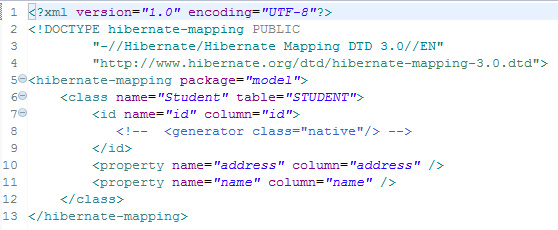
**2. This part contains Hibernate and JDBC coding differences between this project and project ' ServletJSPJDBCFullApp'.**

Please note we are now going to study the additional files in this project which are not there in project ' ServletJSPJDBCFullApp' as that is JDBC based project.

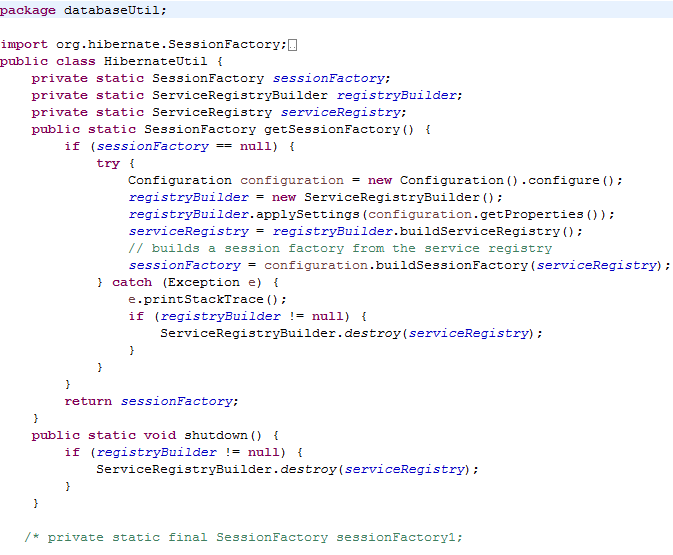
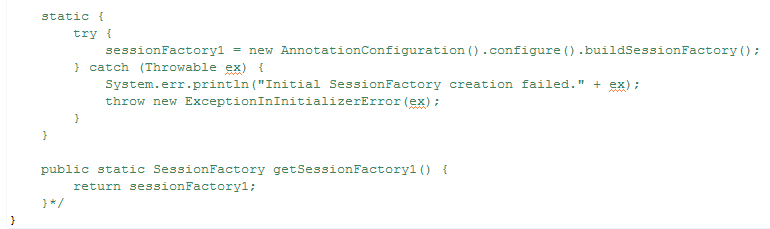
i) Hibernate configuration file viz. hibernate.cfg.xml in resources folder:



ii) Hibernate mapping file viz. Student.hbm.xml in model package



iii) HibernateUtil.java file in databaseUtil package which returns us sessionFactory object (this is real time use of 'Factory design pattern')

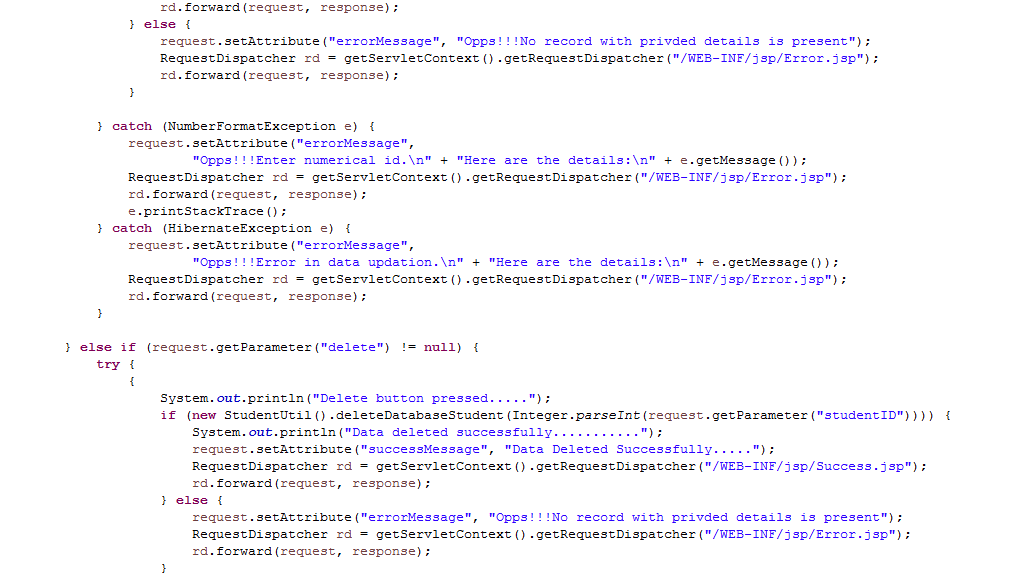
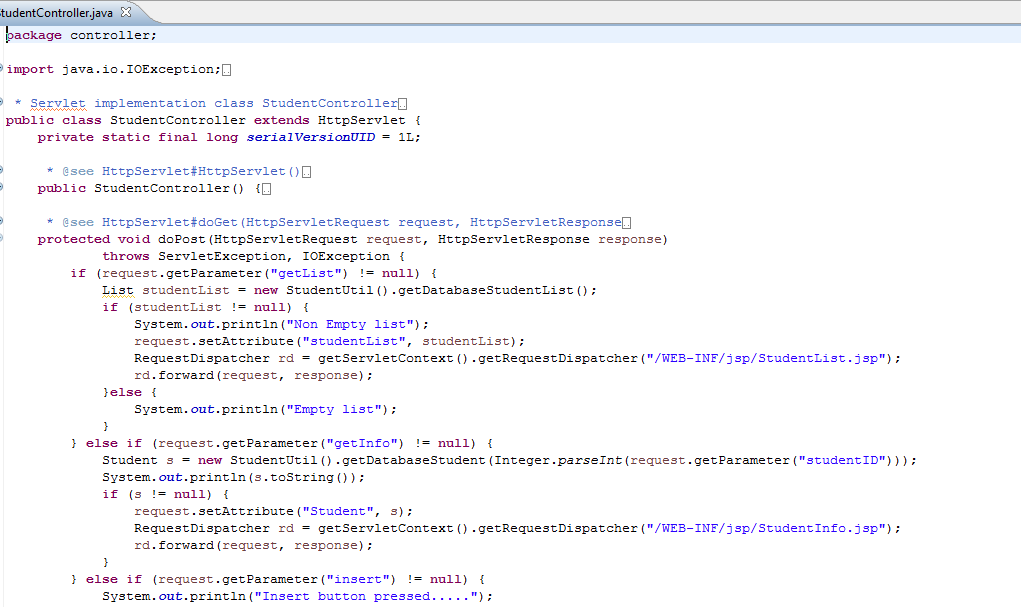
 

**Section II**

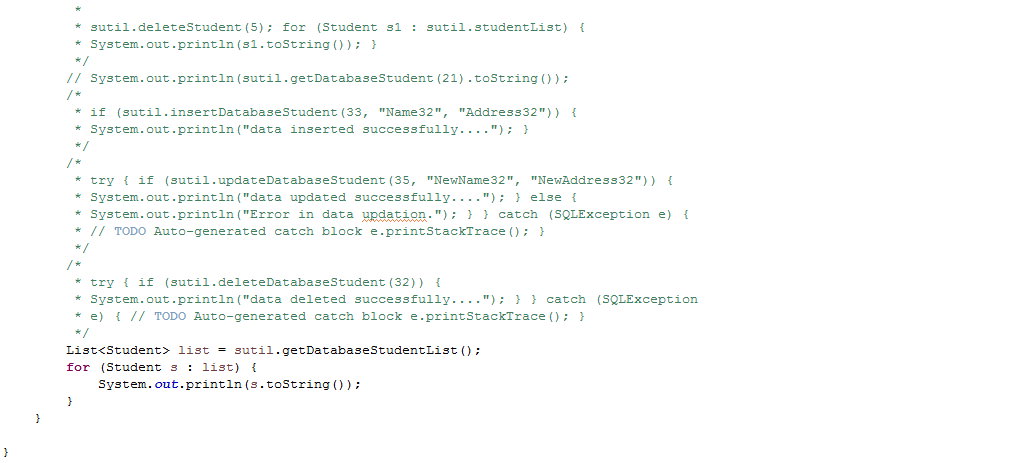
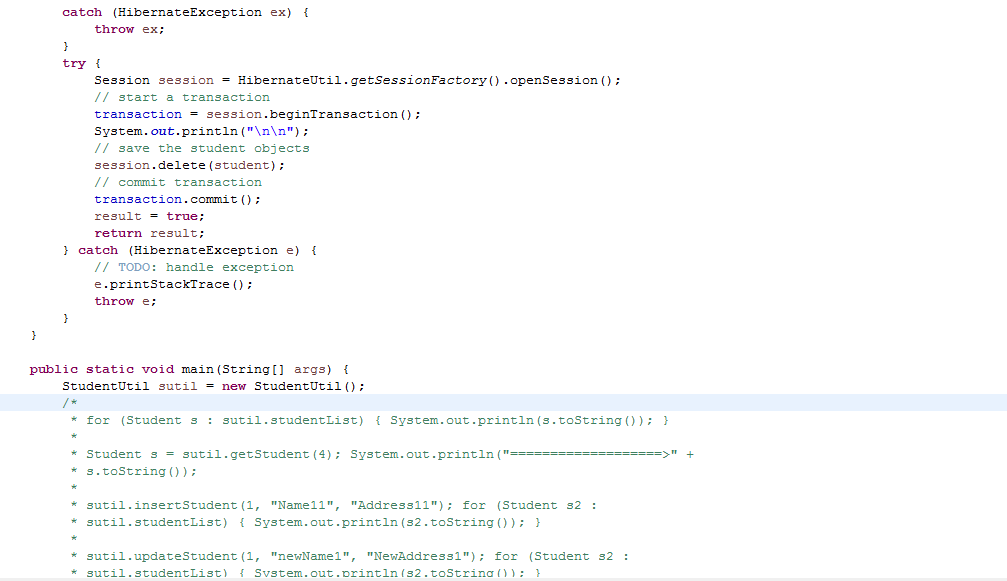
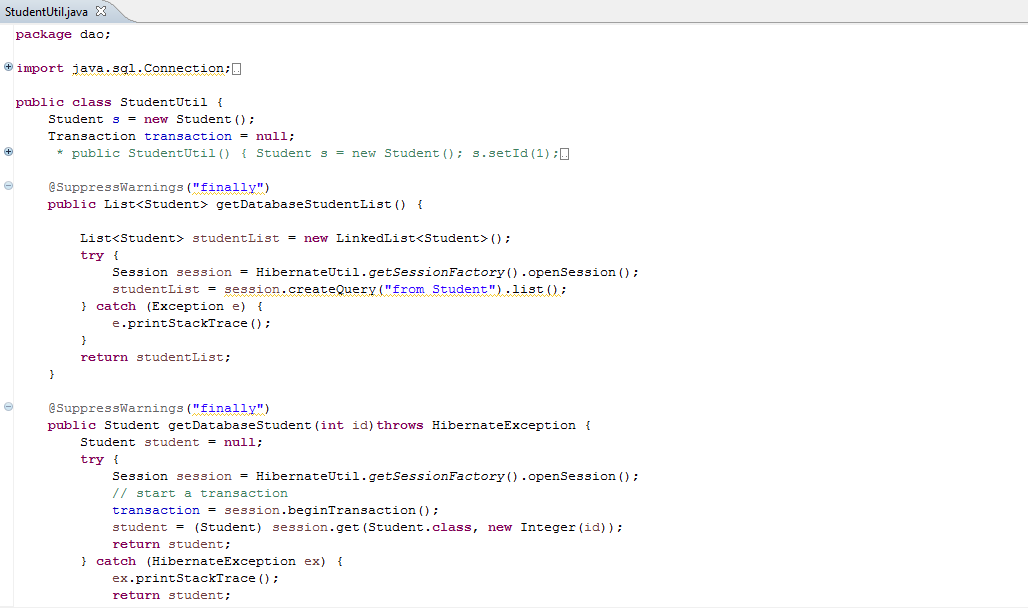
**\*\*\*This section contains the common part with project ' ServletJSPJDBCFullApp' so please refer following points which are related to this project and about its explanation please refer ProjectGuide file of project ' ServletJSPJDBCFullApp'.**

Look for the package structure

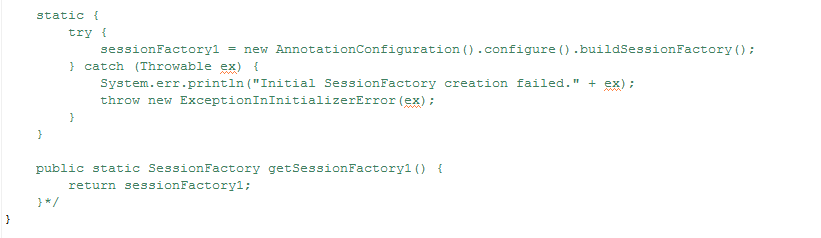
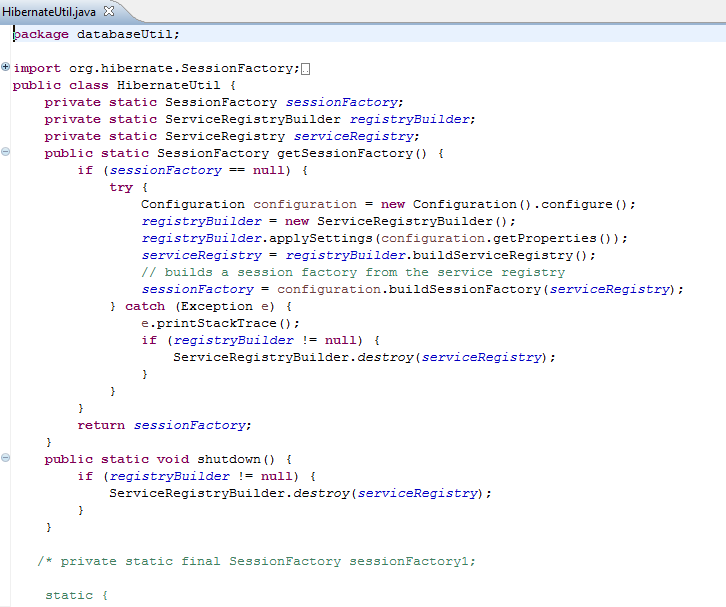
1. There is **controller** package containing servlets acts as contoller of MVC



2. **dao** package containing Data Access Object class responsible for CRUD operations on underlying models classes



3. **databaseUtil** package for hibernate configuration and utility class for database connection in both the ways. Look both the files very carefully and go through the difference among them.

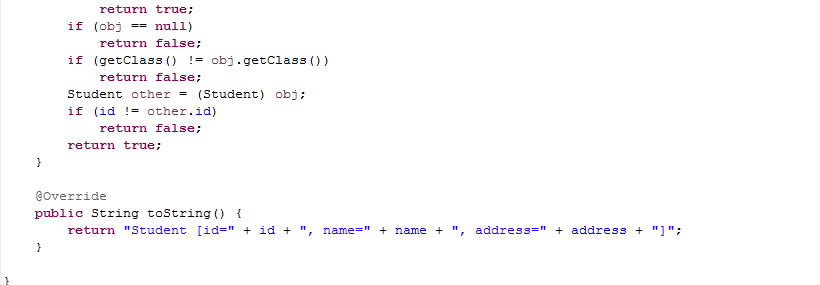


4. **filters** package responsible for containing PerformanceFilter.java file. This filter as the name suggest, responsible for calculating time between request and respective response. This class also contains logging operations. (\*We have explained these files in detail in last section (viz. logging mechanism) of this guide.)

5. **listener** package containing class implementing Context Listener interface.

(\*We have explained these files in detail in last section (viz. logging mechanism) of this guide.)

6. **model** package contains model classes and their respective .hbm.xml file



**\*\*\*\*VVIMP: Before moving further, please go through package structure of this project along with use of each package very carefully. Just mug up it very well.**

**We are going to use almost same package structure in all above frameworks project viz. Struts2 and Spring.**

**How to create this project??**

a) This is maven web project.

(Please refer 'CreateMavenWebProjectWithEclipse.docx' file to get idea about creating such project.)

b) Add require maven dependencies

c) Create and place packages and files in the respective locations and shown. Pay special attention to log4j.xml and jsp folder in src/main/webapp folder.

**Files require more attention**

1. How we form package structure

2. Mapping in web.xml file(Very important)

3. Location and details in log4.xml file

4. various files in jsp folder

5. Files in controller package to study the handling of the request.

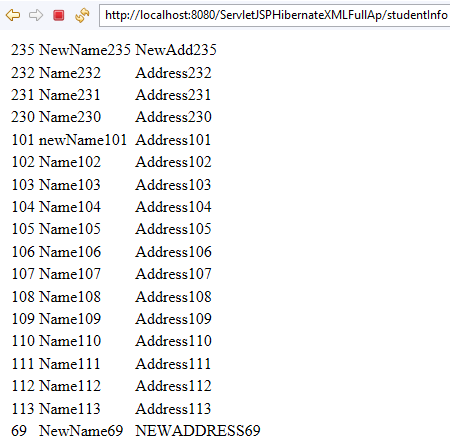
**To run the application:**

Right click on the project and Run As-> Run on server

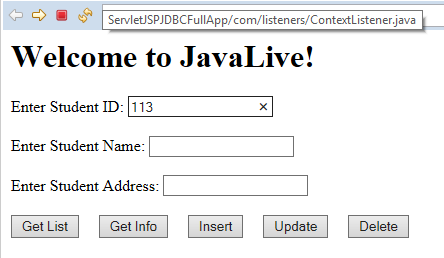
When the application runs it will look as below:



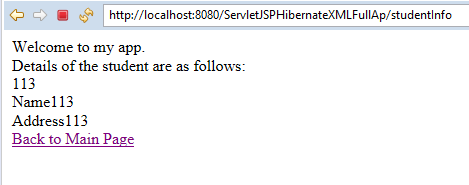
Click 'Get List" button, output is as below



Click back arrow at top left.

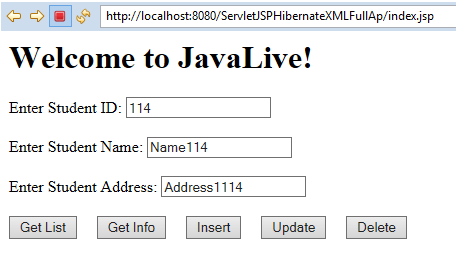


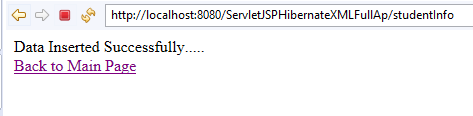
Now enter the given value and click 'Get Info' button. It will shows following details.



Now click 'Back on Main Page' link

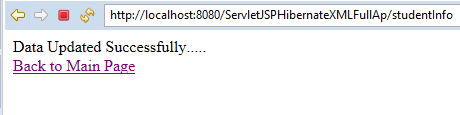
Now enter the values as below to add new entry and click 'Insert' button.





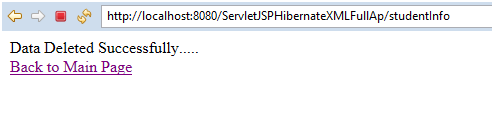
Now click 'Back to Main Page' link.

Now make change in the values of recently created record and click 'Update' button.

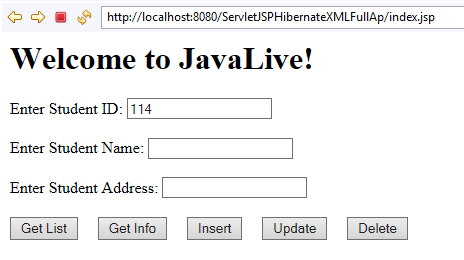
Click 'Back to Main Page' link.

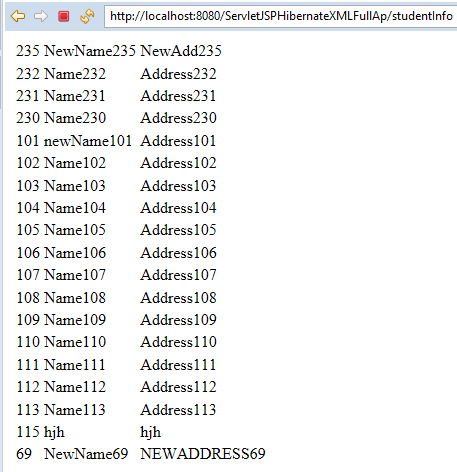
Enter primary key value and click 'Delete' button. It shows following screens:

Click 'Back to Main Page' link

Cross verify whether the record is deleted by clicking 'Get List' button:





Look here is no record with id as '114'.

Also go through console output as follows:

**INFO 2020-09-11 12:03:05,308 [http-nio-8080-exec-5] org.hibernate.annotations.common.Version - HCANN000001: Hibernate Commons Annotations {4.0.2.Final}**

**INFO 2020-09-11 12:03:05,351 [http-nio-8080-exec-5] org.hibernate.Version - HHH000412: Hibernate Core {4.2.2.Final}**

**INFO 2020-09-11 12:03:05,364 [http-nio-8080-exec-5] org.hibernate.cfg.Environment - HHH000206: hibernate.properties not found**

**INFO 2020-09-11 12:03:05,373 [http-nio-8080-exec-5] org.hibernate.cfg.Environment - HHH000021: Bytecode provider name : javassist**

**INFO 2020-09-11 12:03:05,488 [http-nio-8080-exec-5] org.hibernate.cfg.Configuration - HHH000043: Configuring from resource: /hibernate.cfg.xml**

**INFO 2020-09-11 12:03:05,489 [http-nio-8080-exec-5] org.hibernate.cfg.Configuration - HHH000040: Configuration resource: /hibernate.cfg.xml**

**INFO 2020-09-11 12:03:05,748 [http-nio-8080-exec-5] org.hibernate.cfg.Configuration - HHH000221: Reading mappings from resource: model/Student.hbm.xml**

**INFO 2020-09-11 12:03:06,399 [http-nio-8080-exec-5] org.hibernate.cfg.Configuration - HHH000041: Configured SessionFactory: null**

**INFO 2020-09-11 12:03:06,691 [http-nio-8080-exec-5] org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl - HHH000402: Using Hibernate built-in connection pool (not for production use!)**

**INFO 2020-09-11 12:03:06,715 [http-nio-8080-exec-5] org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl - HHH000115: Hibernate connection pool size: 1**

**INFO 2020-09-11 12:03:06,716 [http-nio-8080-exec-5] org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl - HHH000006: Autocommit mode: false**

**INFO 2020-09-11 12:03:06,719 [http-nio-8080-exec-5] org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl - HHH000401: using driver [com.mysql.cj.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/test?useSSL=false]**

**INFO 2020-09-11 12:03:06,720 [http-nio-8080-exec-5] org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl - HHH000046: Connection properties: {user=root, password=\*\*\*\*}**

**INFO 2020-09-11 12:03:08,152 [http-nio-8080-exec-5] org.hibernate.dialect.Dialect - HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect**

**INFO 2020-09-11 12:03:08,570 [http-nio-8080-exec-5] org.hibernate.engine.transaction.internal.TransactionFactoryInitiator - HHH000399: Using default transaction strategy (direct JDBC transactions)**

**INFO 2020-09-11 12:03:08,606 [http-nio-8080-exec-5] org.hibernate.hql.internal.ast.ASTQueryTranslatorFactory - HHH000397: Using ASTQueryTranslatorFactory**

**Hibernate: select student0\_.id as id1\_0\_, student0\_.address as address2\_0\_, student0\_.name as name3\_0\_ from STUDENT student0\_**

**Non Empty list**

**Hibernate: select student0\_.id as id1\_0\_0\_, student0\_.address as address2\_0\_0\_, student0\_.name as name3\_0\_0\_ from STUDENT student0\_ where student0\_.id=?**

**Student [id=113, name=Name113, address=Address113]**

**Insert button pressed.....**

**Hibernate: insert into STUDENT (address, name, id) values (?, ?, ?)**

**Data inserted successfully...........**

**Update button pressed.....**

**Hibernate: update STUDENT set address=?, name=? where id=?**

**Data updated successfully...........**

**Delete button pressed.....**

**Hibernate: select student0\_.id as id1\_0\_0\_, student0\_.address as address2\_0\_0\_, student0\_.name as name3\_0\_0\_ from STUDENT student0\_ where student0\_.id=?**

**Hibernate: select student\_.id, student\_.address as address2\_0\_, student\_.name as name3\_0\_ from STUDENT student\_ where student\_.id=?**

**Hibernate: delete from STUDENT where id=?**

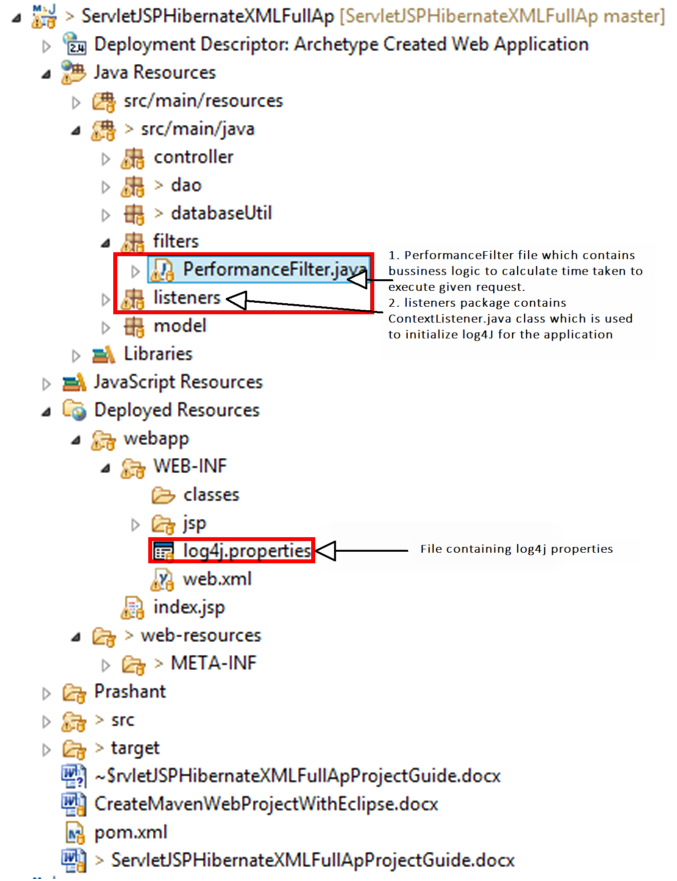
**Data deleted successfully...........**

**Hibernate: select student0\_.id as id1\_0\_, student0\_.address as address2\_0\_, student0\_.name as name3\_0\_ from STUDENT student0\_**

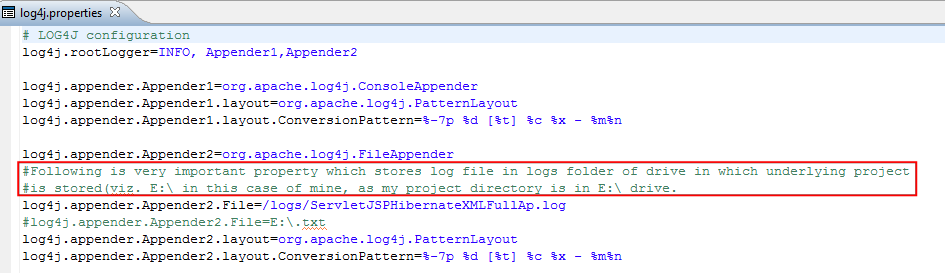
**Non Empty list**

**Note the logging mechanism in this application:**

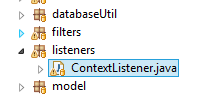
Various files involved in the logging mechanism are as follows:



1. Log4J.properties file



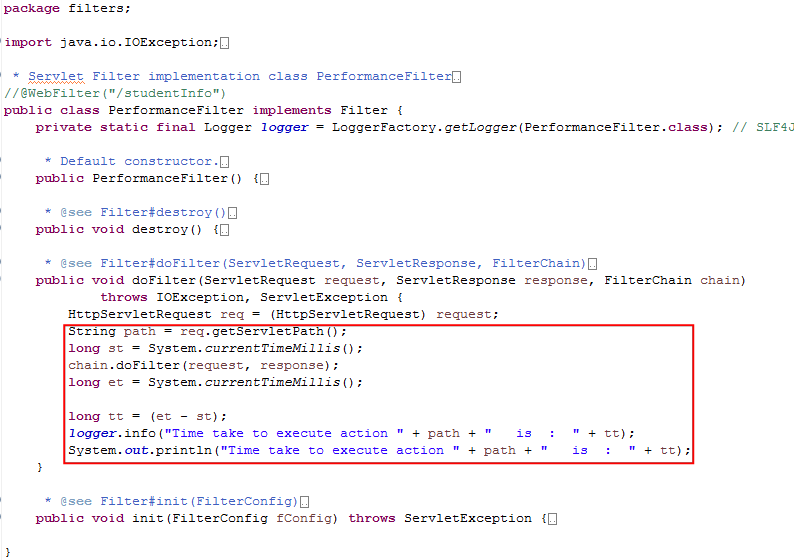
2. listeners.ContextListener class containing logic to configure log4j mechanism for this application:



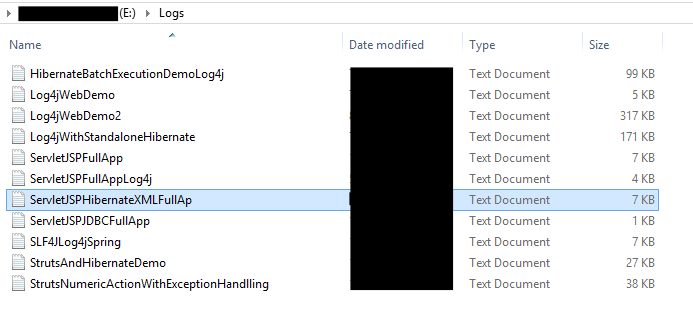


Please refer link <http://mysupport.download.microfocus.com/SSO/online_help/IDOL/Interfaces/Find/11.6/Guides/html/English/admin/Content/ConfigSystemProps/ConfigureLogLocation.htm> for more details.

3. filters.PerformanceFilter class which contains logic to calculate time taken to execute each request:



4. Location of created log file in the system is as follows:



5. Contents of the above file are as below:

C:\Users\Administrator\Documents\DonationCoder\ScreenshotCaptor\Screenshots\Screenshot - 10_26_2021 , 3_00_01 PM.png

**\*\*\*Please note all above logging mechanism very carefully to understand role of logging in live project.**

Thus in this application we have covered almost all concepts in servlets and JSP technology viz. MVC design pattern, servlet as controller, filters, listeners, model, error handling using JSP, database handling using Hibernate, logging mechanism etc.