**LIBRARY MANAGEMENT SYSTEM**

**SUMMARY**

**This project consists of the following:**

1. **com.nhce.adobe.bean**

|  |
| --- |
| **AuthorBean.java :**  **int** authorCode;  String authorName; |
| **BookBean.java**  String isbn;  String bookName;  AuthorBean author;  **char** bookType; // Category of Book( Can have ‘G’ or ‘T’  **float** cost; |

1. **com.nhce.adobe.dao**

|  |
| --- |
| **Bookdao.java**  BookBean fetchBook(String isbn)  createBean(BookBean bookBean)  string generateId(String bookName): This method should contain the necessary code to create a new isbn.  addAuthor(String authorname)  Autoincrement authorcode and add to table  HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  This method processes the Client’s request. It accesses a request parameter called ‘operation’ and based on the value of that parameter, it’ll decide whether to access the addBook (HttpServletRequest request) or viewBook(String isbn) method  · [Note: The name of the submit button in ‘AddBook.html’ and ‘ViewBook.html’ is ‘operation’ and the Servlet uses this to determine what operation needs to be done]  · If the operation parameter value is ‘AddBooks’ then the following things needs to be done  1. addBook(request) method is called  2. If return value equals “SUCCESS” the servlet should redirect to “Menu.html”  3. If return value equals “FAILURE” the servlet should redirect to “Failure.html”  4. If return value equals “INVALID” the servlet should redirect to “Invalid.html”  · If the operation parameter value is ‘Search’ then the following things needs to be done  1. viewBook(String isbn) method will be invoked  2. If the return value is null the servlet should redirect to “Invalid.html”  3. Else the bean should be set as an attribute to the request object and should forward the control to “View.jsp”. Use requestdispatcher.forward |

1. **com.nhce.adobe.util**

|  |
| --- |
| **DBUtil.java** |

**4. It contains the following views:**

1. **index.html:** The user can select one of the following options: 1. ADD BOOKS 2.VIEW BOOKS.
2. **Menu.html:** Here the user can add the books.
3. **View.jsp**: It is used to search the books based on ISBN value.
4. **Failure.html:** The user gets this view when adding the book is unsuccessful.
5. **Invalid.html:** The user gets this view when the ISBN of the book he is looking for is invalid.

**Index.html**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Hello!!</title>

</head>

<body>

<form action=*"Menu.html"* method=*"get"*>

<input type=*"submit"* value=*"ADD BOOKS"*>

</form>

<br>

<form action=*"View.jsp"* method=*"get"*>

<input type=*"submit"* value=*"VIEW BOOKS"*>

</form>

</body>

</html>

****

**Fig 1:**index.html

**Menu.html**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<h1>ADD BOOKS</h1>

<form action=*"http://localhost:8080/library/Bookdao"* method=*"get"*>

<label>Book Name: </label><input type=*"text"* name=*"bookName"*><br>

<label>Author Name: </label><input type=*"text"* name=*"authorName"*><br>

<label>Book Type: </label><input type=*"text"* name=*"bookType"*><br>

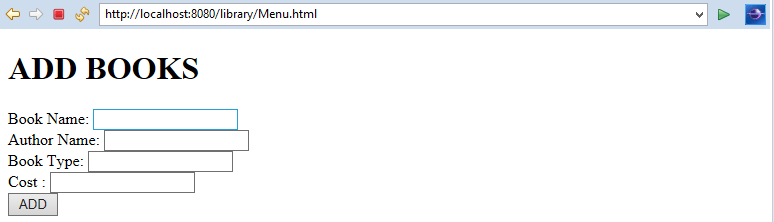
<label>Cost : </label><input type=*"text"* name=*"cost"*><br>

<input type=*"submit"* value=*"ADD"*><br>

</form>

</body>

</html>

****

**Fig 2:** menu.html

**View.jsp**

<%@ page language=*"java"* contentType=*"text/html; charset=ISO-8859-1"*

pageEncoding=*"ISO-8859-1"*%>

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv=*"Content-Type"* content=*"text/html; charset=ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<form action=*"http://localhost:8080/library/Viewdao"*>

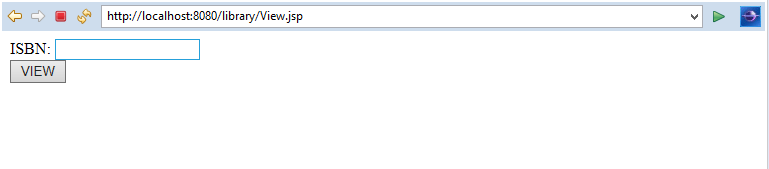
<label>ISBN: </label><input type=*"text"* name=*"isbn"*><br>

<input type=*"submit"* value=*"VIEW"*><br>

</form>

</body>

</html>

****

**Fig 3:** view.jsp

**Failure.html**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

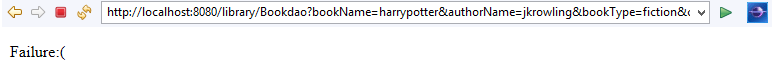
</head>

<body>

<p>Failure:(</p>

</body>

</html>

****

**Fig 4**: failure.html

**Invalid.html**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

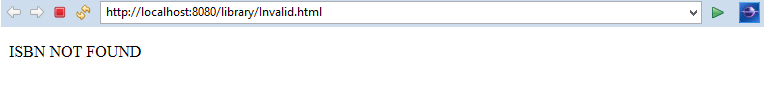
</head>

<body>

<p>ISBN NOT FOUND</p>

</body>

</html>

****

**Fig 5:** invalid.html

**AuthorBean.java**

**package** com.nhce.adobe.bean;

**public** **class** AuthorBean {

**int** authorCode;

**public** **int** getAuthorCode() {

**return** authorCode;

}

**public** **void** setAuthorCode(**int** authorCode) {

**this**.authorCode = authorCode;

}

**public** String getAuthorName() {

**return** authorName;

}

**public** **void** setAuthorName(String authorName) {

**this**.authorName = authorName;

}

String authorName;

}

**BookBean.java**

**package** com.nhce.adobe.bean;

**public** **class** BookBean {

String isbn;

**public** String getIsbn() {

**return** isbn;

}

**public** **void** setIsbn(String isbn) {

**this**.isbn = isbn;

}

**public** String getBookName() {

**return** bookName;

}

**public** **void** setBookName(String bookName) {

**this**.bookName = bookName;

}

**public** AuthorBean getAuthor() {

**return** author;

}

**public** **void** setAuthor(AuthorBean author) {

**this**.author = author;

}

**public** String getBookType() {

**return** bookType;

}

**public** **void** setBookType(String bookType) {

**this**.bookType = bookType;

}

**public** **float** getCost() {

**return** cost;

}

**public** **void** setCost(**float** cost) {

**this**.cost = cost;

}

String bookName;

AuthorBean author;

String bookType;

**float** cost;

}

**DBUtil.java**

This code is used to connect to the database. If it cannot connect to the database, it throws ClassNotFoundException or SqlException based on the error. Here the name of the database we are using is “library”. It contains two tables author and bookinfo. The author table contains aid and author name. The bookinfo table contains book ISBN, author id, book type, book name and cost.

**package** com.nhce.adobe.util;

**import** java.sql.\*;

**public** **class** DBUtil {

**public** Connection createConnection()

{

Connection con=**null**;

**try**{

Class.*forName*("com.mysql.jdbc.Driver");

//name of db:library

//create author table:aid primary key auto-increment,aname

//create bookinfo table:isbn primary key,aid foreign key,bookname,booktype,cost

con=DriverManager.*getConnection*("jdbc:mysql://localhost:3306/library","root","ROOT123");

System.***out***.println("Connected Successfully");

}

**catch**(ClassNotFoundException e)

{

e.printStackTrace();

}

**catch**(SQLException e)

{

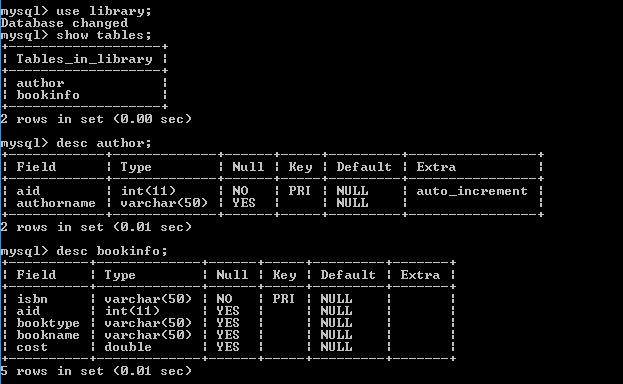
e.printStackTrace();

}

**return** con;

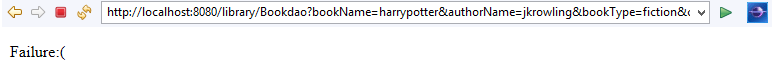
}

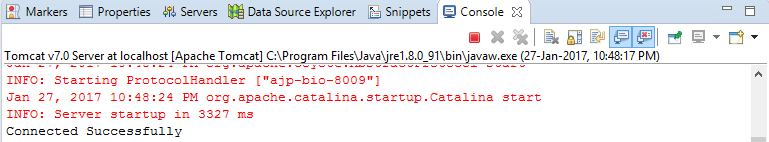
}

****

**Fig 6:** Description of author and bookinfo table.

If it cannot connect to the database successfully, it redirects to failure.html.

**Fig 7:** Failure to connect to the database.

****

**Fig 8:** Console view on successful connection to database

**BookDao.java**

It comes here when the user wants to add a book. It checks if the author is already present. If the author info is not present in the database, it adds the author information into the author table. Here the author id is increments automatically on adding a value in the table. Next, the book information is added into the bookinfo table by retrieving the values from the form.

package com.nhce.adobe.dao;

import com.nhce.adobe.util.\*;

import java.sql.\*;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.nhce.adobe.bean.AuthorBean;

import com.nhce.adobe.bean.BookBean;

/\*\*

\* Servlet implementation class Bookdao

\*/

/\* It generates the book ISBN by retrieving the first two letters from the bookname and concatenating it with the hash code of the book name\*/

public String stringGenerateId(String bookName)

{ String sub=bookName.substring(0,2);

int code=bookName.hashCode();

String code1=String.valueOf(code);

sub=sub.concat(code1);

return sub;

}

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

BookBean b=new BookBean();

DBUtil d=new DBUtil();

String bname=request.getParameter("bookName");

b.setBookName(bname);

String booktype=request.getParameter("bookType");

b.setBookType(booktype);

String bid=stringGenerateId(request.getParameter("bookName"));

b.setIsbn(bid);

String cost=request.getParameter("cost");

float c=Float.parseFloat(cost);

b.setCost(c);

AuthorBean a = new AuthorBean();

String aname= request.getParameter("authorName");

//creating connection to the database

Connection con=d.createConnection();

Statement st=null;

Statement st1=null;

ResultSet rs=null;

ResultSet rs1=null;

int aid=0;

//checking author table if author is already present,if present get the author id else generate new id and insert it into table.

try{

st=con.createStatement();

rs=st.executeQuery("select aid from author where authorname='"+aname+"'");

//while(rs.next()){

if(!rs.isBeforeFirst() )//if author is not present in db,insert the values inside db

{

System.out.println("Author is new");

//int id=aname.hashCode();

//here aid is auto-incremented,will be stored automatically

st1=con.createStatement();

st1.executeUpdate("insert into author(authorname)values('"+aname+"')");

System.out.println("Inserted");

//get the aid of the author whose values was recently put into the db

//in order to store it in author and book bean respectively

//rs.close();

rs1=st.executeQuery("select \* from author where authorname='"+aname+"'");

while(rs1.next()){

aid=rs1.getInt("aid");

System.out.println(aid);

}

a.setAuthorCode(aid);

a.setAuthorName(aname);

}

else//if author is present get values to put in author and book bean objects

{

System.out.println("Author is old");

while(rs.next()){

aid=rs.getInt("aid");}

System.out.println(aid);

a.setAuthorCode(aid);

a.setAuthorName(aname);

}

//System.out.println("success 1");

}

catch(Exception e){e.printStackTrace();}

b.setAuthor(a);

//to insert into bookinfo table

try{

int authorid=aid;

System.out.println(authorid);

st.executeUpdate("insert into bookinfo values('"+bid+"','"+aid+"','"+booktype+"','"+bname+"','"+cost+"')");

//System.out.println("Success");

PrintWriter pw=response.getWriter();

pw.println("BookName: "+bname+"\n");

pw.println("AuthorName: "+aname+"\n");

pw.println("Book ISBN: "+bid+"\n");

pw.println("Book Type: "+booktype+"\n");

pw.println("Cost: "+cost+"\n");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

//servlet2 is the url-pattern of the second servlet

rd.include(request, response);//method may be include or forward

}

catch(Exception e){

//System.out.println("Failure");

//since its a failure, we have to forward request to forward.html

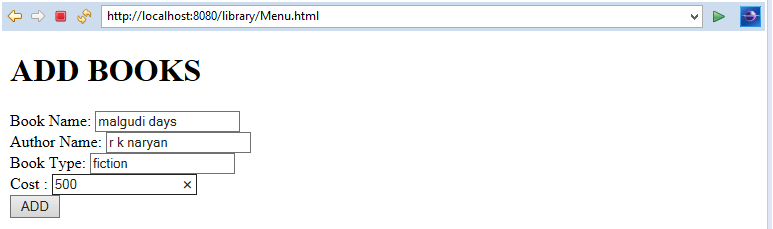
RequestDispatcher rd=request.getRequestDispatcher("Failure.html");

//servlet2 is the url-pattern of the second servlet

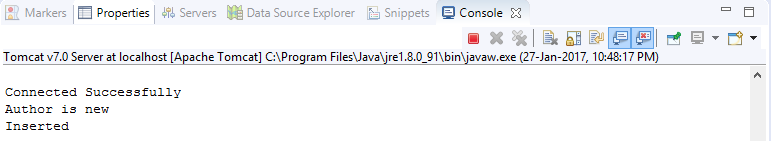
rd.forward(request, response);//method may be include or forward

e.printStackTrace();}

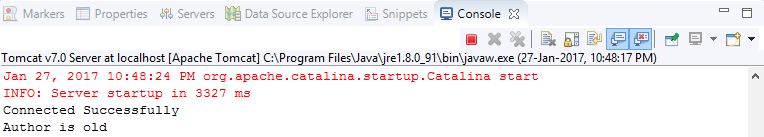
}



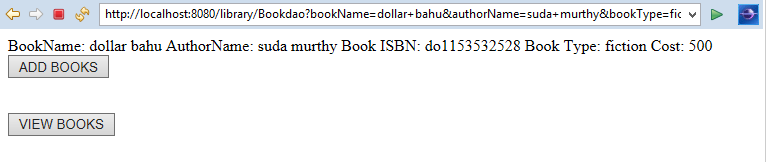
**Fig 9:** Adding a book

****

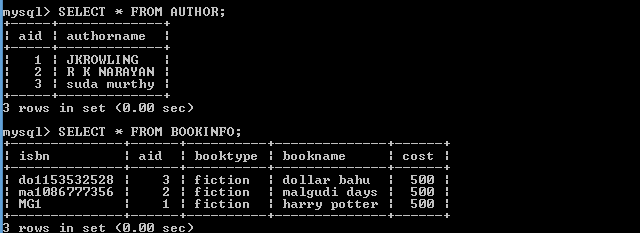
**Fig 10:** Console view if the author is not present in the database



**Fig 11:** Console view if the author is already present in the database



**Fig 12:** On adding new author, it redirects to menu.html



**Fig 13:** Contents of author and bookinfo table

**ViewDao.java**

In order to search a book based on ISBN, the user enters the ISBN number. If the book with the ISBN number is present, the data for that particular book is retrieved and is shown to the user else it redirects to invalid.html page.

**package com.nhce.adobe.dao;**

**import java.io.IOException;**

**import java.io.PrintWriter;**

**import java.sql.Connection;**

**import java.sql.ResultSet;**

**import java.sql.Statement;**

**import javax.servlet.RequestDispatcher;**

**import javax.servlet.ServletException;**

**import javax.servlet.annotation.WebServlet;**

**import javax.servlet.http.HttpServlet;**

**import javax.servlet.http.HttpServletRequest;**

**import javax.servlet.http.HttpServletResponse;**

**import com.nhce.adobe.util.DBUtil;**

protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

DBUtil d=new DBUtil();

PrintWriter pw=response.getWriter();

Connection con=d.createConnection();

String isbn=request.getParameter("isbn");

Statement st=null;

Statement st3=null;

ResultSet rs2=null;

ResultSet rs3=null;

String bookname=null;

String booktype=null;

String aname=null;

double cost=0;

int aid=0;

try{

st=con.createStatement();

st3=con.createStatement();

rs2=st.executeQuery("select \* from bookinfo where isbn='"+isbn+"'");

if(!rs2.isBeforeFirst() )

{

}

else{

while(rs2.next())

{

System.out.println("inside while");

bookname=rs2.getString("bookname");

booktype=rs2.getString("booktype");

cost=rs2.getDouble("cost");

aid=rs2.getInt("aid");

rs3=st3.executeQuery("select \* from author where aid='"+aid+"'");

while(rs3.next())

{

System.out.println(rs3.getString("authorname"));

aname=rs3.getString("authorname");

}

}}

pw.println("BookName: "+bookname+"\n");

pw.println("AuthorName: "+aname+"\n");

pw.println("Book ISBN: "+isbn+"\n");

pw.println("Book Type: "+booktype+"\n");

pw.println("Cost: "+cost+"\n");

RequestDispatcher rd=request.getRequestDispatcher("index.html");

rd.include(request, response);

}

catch(Exception e)

{

e.printStackTrace();

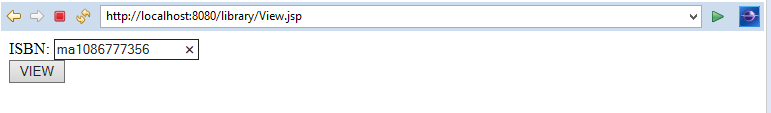
pw.println("Not valid isbn");

RequestDispatcher rd=request.getRequestDispatcher("Invalid.html");

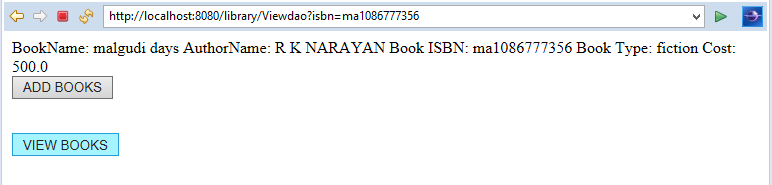
rd.include(request, response);

}

}



**Fig 14:** Enter the ISBN to view the details of the book



**Fig 15:** The details of the book is displayed on menu.html page

If the ISBN is not found, it redirects to invalid.html page.