2020

Proof of working code for Enterprise Programming Coursework Project

MANCHESTER METROPOLITAN UNIVERSITY
ANTE TOMICIC

Introduction

In this PDF file, I will be going through each completed section of the Enterprise Programming coursework in Manchester Metropolitan University (MMU) and provide proof of achievement for each completed section along with screenshots and short commentary.

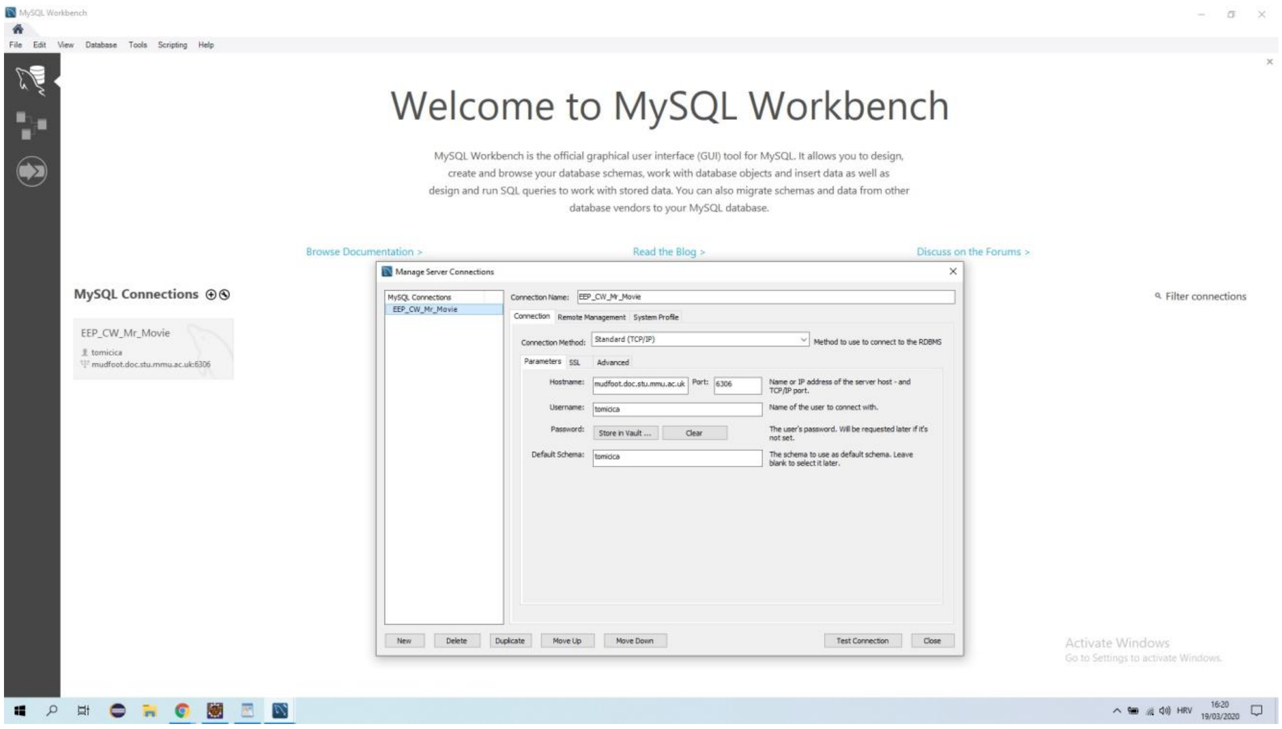
In short, the coursework task is making a dynamic web service using MySQL for the database in MySQL Workbench, Java for the back-end and AJAX, JavaScript, jQuery, HTML, CSS for the front-end, all inside Eclipse IDE using Tomcat 9.0 for the local server and Google Cloud Platform for cloud deployment. Along with an API, RESTful, SOAP and WSDL implementation and implementation of sound Software Engineering (SE) practices.

With each chapter I will go through main sections of the coursework and provide suitable explanation for each section that has been completed successfully.

However, I have not attempted all tasks. Task such as SOAP/WSDL and REST have not been completed nor tried.

Database implementation

The database used in the development process was not local, rather it is stored at MMU servers with access allowed to the database. Picture 1. shows the connection to the database server and Picture 2. shows a “SELECT * FROM films;” statement executed as proof of working and finished database.



Welcome to MySQL Workbench

MySQL Workbench is the official graphical user interface (GUI) tool for MySQL. It allows you to design, create and browse your database schemas, work with database objects and insert data as well as design and run SQL queries to work with stored data. You can also migrate schemas and data from other database vendors to your MySQL database.

MySQL Connections

EEP_CW_Mr_Movie
tomicica
mudfoot.doc.stu.mmu.ac.uk:6306

Manage Server Connections

Connection Name: EEP_CW_Mr_Movie

Connection Method: Standard (TCP/IP) — Method to use to connect to the RDBMS

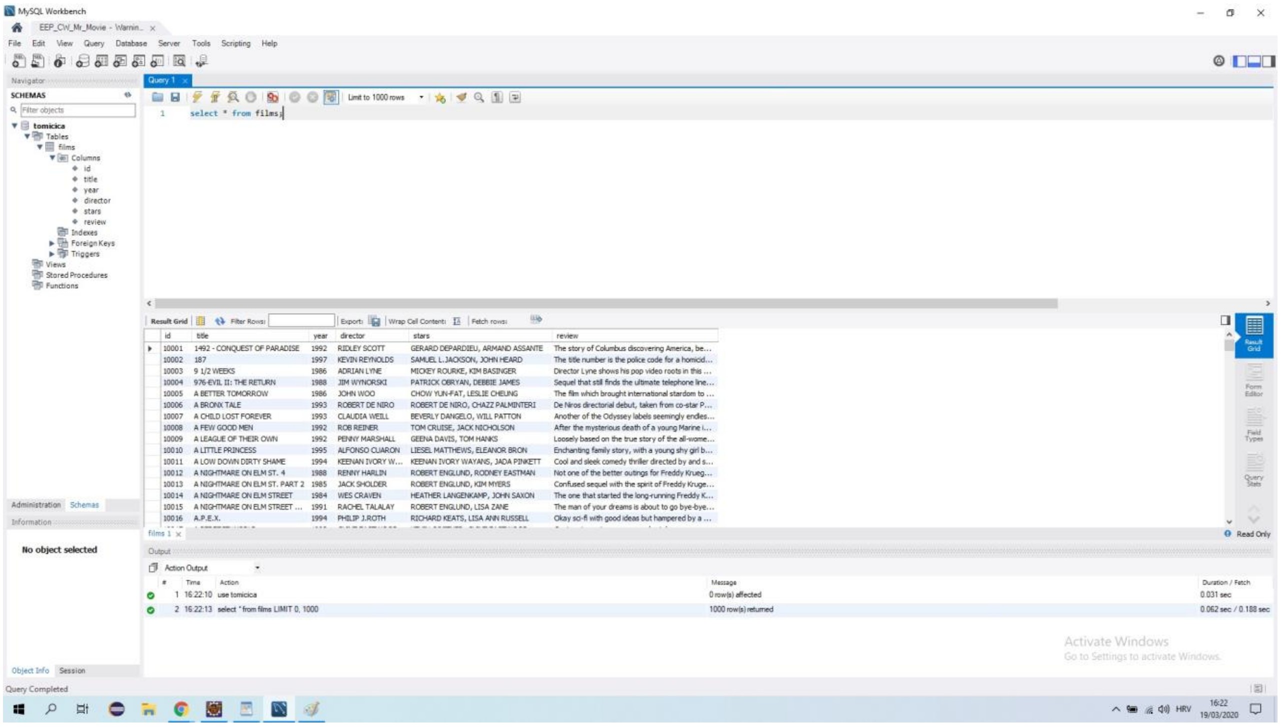
Hostname: mudfoot.doc.stu.mmu.ac.uk Port: 6306 — Name or IP address of the server host - and TCP/IP port.

Username: tomicica — Name of the user to connect with.

Password: — The user's password. Will be requested later if it's not set.

Default Schema: tomicica — The schema to use as default schema. Leave blank to select it later.

Picture 1. MMU server, database connection



```
select * from films;
```

id	title	year	director	stars	review
10001	1492 - CONQUEST OF PARADISE	1992	RIDLEY SCOTT	GERARD DEPARDIEU, ARMAND ASSANTE	The story of Columbus discovering America, be...
10002	187	1997	KEVIN REYNOLDS	SAMUEL L.JACKSON, JOHN HEARD	The title number is the police code for a homicid...
10003	9 1/2 WEEKS	1986	ADRIAN LYNE	MICKEY ROURKE, KIM BASINGER	Director Lyne shows his pop video roots in this ...
10004	976-EVIL II: THE RETURN	1988	JIM WYNORSKI	PATRICK OBRYAN, DEBBIE JAMES	Sequel that still finds the ultimate telephone line...
10005	A BETTER TOMORROW	1986	JOHN WOO	CHOW YUN-FAT, LESLIE CHEUNG	The film which brought international stardom to ...
10006	A BRONX TALE	1993	ROBERT DE NIRO	ROBERT DE NIRO, CHAZZ PALMINTERI	De Niros directorial debut, taken from co-star P...
10007	A CHILD LOST FOREVER	1993	CLAUDIA WEILL	BEVERLY DANGELO, WILL PATTON	Another of the Odyssey labels seemingly endles...
10008	A FEW GOOD MEN	1992	ROB REINER	TOM CRUISE, JACK NICHOLSON	After the mysterious death of a young Marine i...
10009	A LEAGUE OF THEIR OWN	1992	PENNY MARSHALL	GEENA DAVIS, TOM HANKS	Loosely based on the true story of the all-wome...
10010	A LITTLE PRINCESS	1995	ALFONSO CUARON	LIESEL MATTHEWS, ELEANOR BRON	Enchanting family story, with a young shy girl b...
10011	A LOW DOWN DIRTY SHAME	1994	KEENAN IVORY W...	KEENAN IVORY WAYANS, JADA PINKETT	Cool and sleek comedy thriller directed by and s...
10012	A NIGHTMARE ON ELM ST. 4	1988	RENNY HARLIN	ROBERT ENGLUND, RODNEY EASTMAN	Not one of the better outings for Freddy Krueg...
10013	A NIGHTMARE ON ELM ST. PART 2	1985	JACK SHOLDER	ROBERT ENGLUND, KIM MYERS	Confused sequel with the spirit of Freddy Kruge...
10014	A NIGHTMARE ON ELM STREET	1984	WES CRAVEN	HEATHER LANGENKAMP, JOHN SAXON	The one that started the long-running Freddy K...
10015	A NIGHTMARE ON ELM STREET ...	1991	RACHEL TALALAY	ROBERT ENGLUND, LISA ZANE	The man of your dreams is about to go bye-bye...
10016	A.P.E.X.	1994	PHILIP J.ROTH	RICHARD KEATS, LISA ANN RUSSELL	Okay sci-fi with good ideas but hampered by a ...

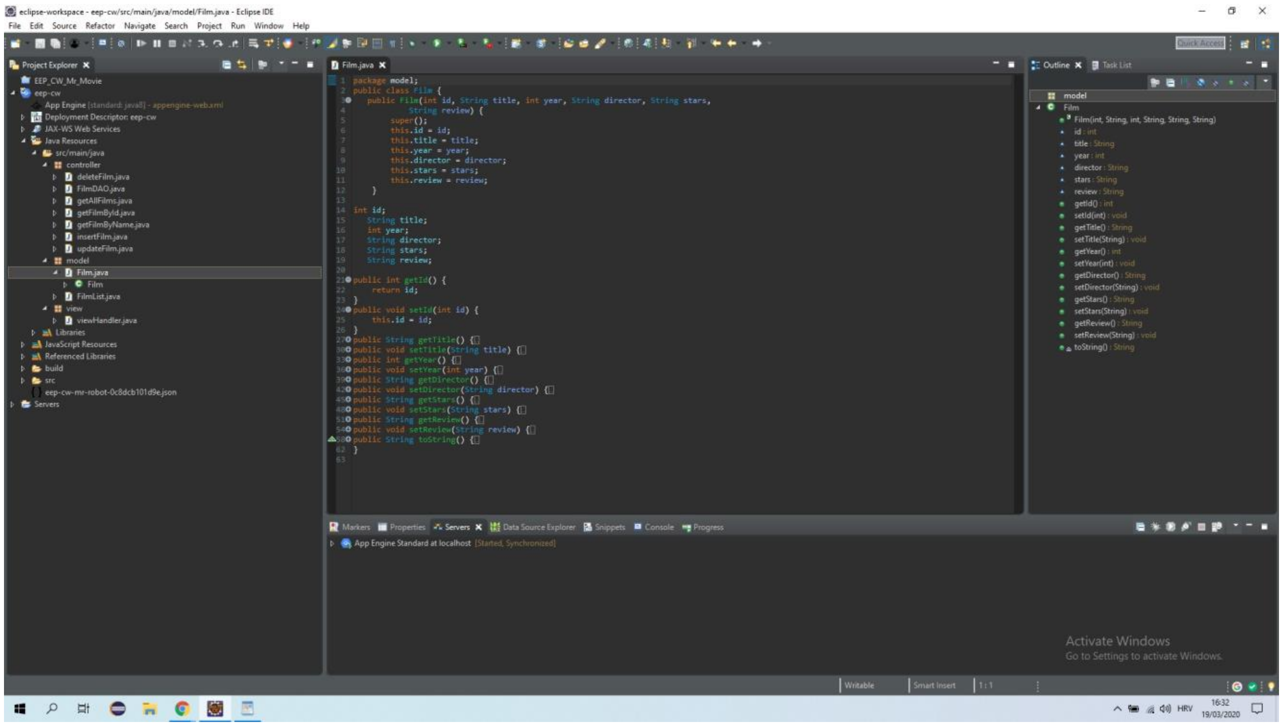
```
#  Time      Action                          Message                 Duration / Fetch
1  16:22:10  use tomicica                    0 row(s) affected       0.031 sec
2  16:22:13  select * from films LIMIT 0, 1000   1000 row(s) returned    0.062 sec / 0.188 sec
```

Picture 2. Proof of working database

Model-View-Controller implementation

In this chapter I will provide proof of Model-View-Controller (MVC) implementation and I will show a single screenshot for the model and for the view, and a couple of screenshots for the controller.

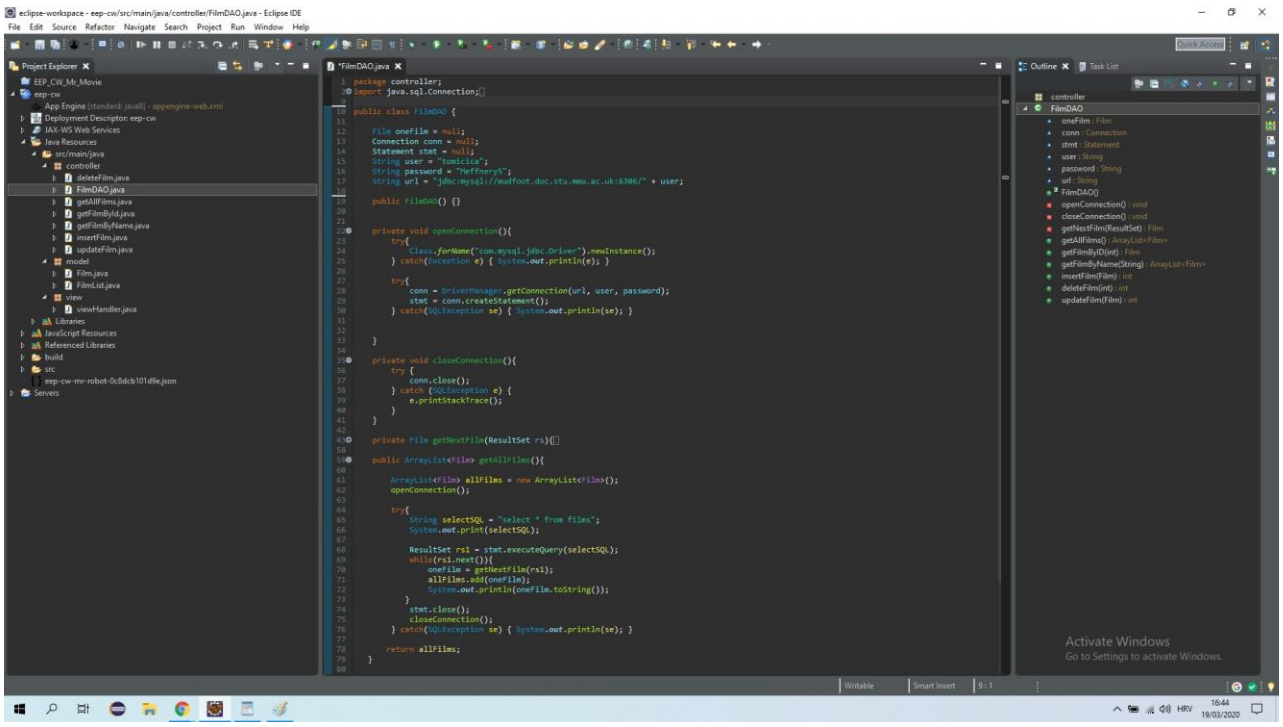
In Picture 3. you can see Eclipse IDE and on the left-hand side in the Project Explorer, the packages used: “Model”, “View”, “Controller”. Each package containing appropriate files and the controller being the only link between the model and the view, along with the code that describes the model and implements a class named Film.



```java
package model;
public class Film {
   public Film(int id, String title, int year, String director, String stars,
            String review) {
        super();
        this.id = id;
        this.title = title;
        this.year = year;
        this.director = director;
        this.stars = stars;
        this.review = review;
    }

int id;
   String title;
   int year;
   String director;
   String stars;
   String review;

public int getId() {
    return id;
}
public void setId(int id) {
    this.id = id;
}
public String getTitle() {
public void setTitle(String title) {
public int getYear() {
public void setYear(int year) {
public String getDirector() {
public void setDirector(String director) {
public String getStars() {
public void setStars(String stars) {
public String getReview() {
public void setReview(String review) {
public String toString() {
}
```

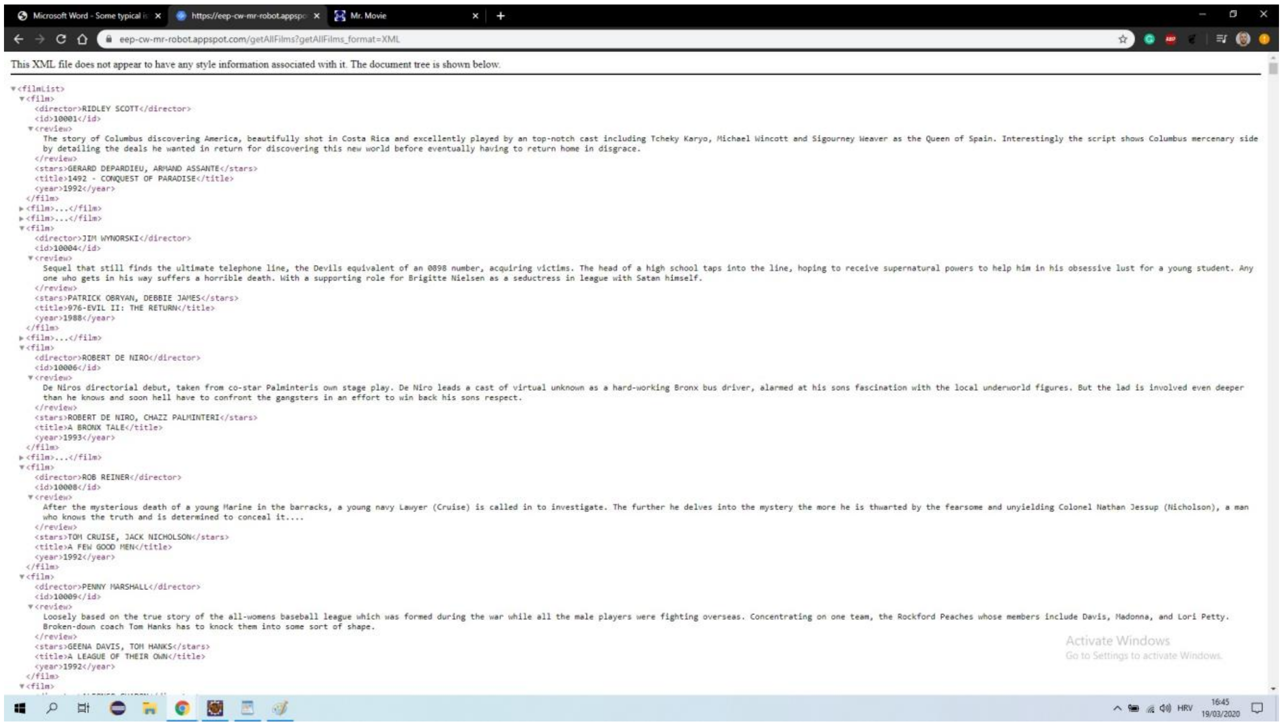
Picture 3.MVC and preview of “Model” code

Picture 4. Shows the Application Programming Interface (API) as the java class called FilmDAO.java (DAO stands for Data Access Object) and is the single class that has the ability to access the model and the database. Picture 4. shows the code for the FilmDAO.java along with implementations of “getAllFilms()”, “openConection()” and “closeConnection” methods. Picture 5. will demonstrate how the API works in the browser and it will do so on the cloud application and “getAllFilms()” method using xml as format.



```java
package controller;
import java.sql.Connection;

public class FilmDAO {

    Film oneFilm = null;
    Connection conn = null;
    Statement stmt = null;
    String user = "tomicica";
    String password = "Meffnery5";
    String url = "jdbc:mysql://mudfoot.doc.stu.mmu.ac.uk:6306/" + user;

    public FilmDAO() {}


    private void openConnection(){
        try{
            Class.forName("com.mysql.jdbc.Driver").newInstance();
        } catch(Exception e) { System.out.println(e); }

        try{
            conn = DriverManager.getConnection(url, user, password);
            stmt = conn.createStatement();
        } catch(SQLException se) { System.out.println(se); }


    }

    private void closeConnection(){
        try {
            conn.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    private Film getNextFilm(ResultSet rs){

    public ArrayList<Film> getAllFilms(){

        ArrayList<Film> allFilms = new ArrayList<Film>();
        openConnection();

        try{
            String selectSQL = "select * from films";
            System.out.print(selectSQL);

            ResultSet rs1 = stmt.executeQuery(selectSQL);
            while(rs1.next()){
                oneFilm = getNextFilm(rs1);
                allFilms.add(oneFilm);
                System.out.println(oneFilm.toString());
            }
            stmt.close();
            closeConnection();
        } catch(SQLException se) { System.out.println(se); }

       return allFilms;
   }
```

Picture 4. FilmDAO.java and appropriate methods implementation

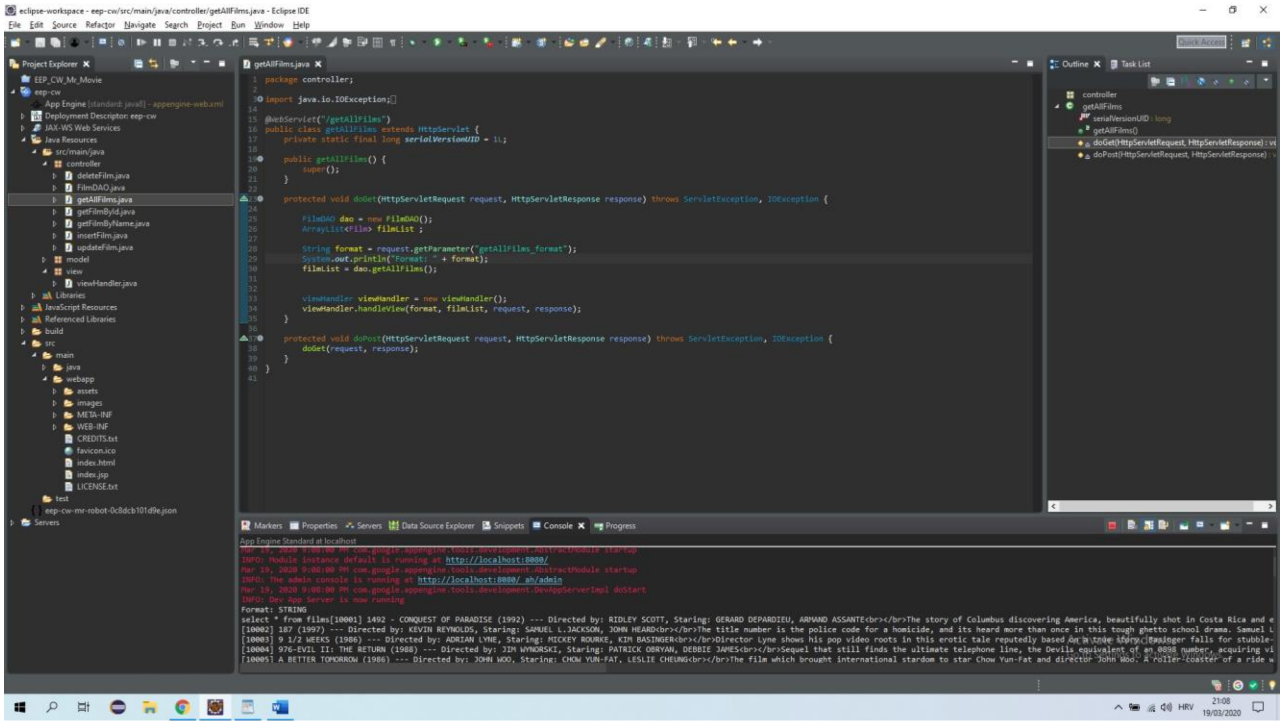


This XML file does not appear to have any style information associated with it. The document tree is shown below.

```xml
▼<filmList>
  ▼<film>
     <director>RIDLEY SCOTT</director>
     <id>10001</id>
    ▼<review>
       The story of Columbus discovering America, beautifully shot in Costa Rica and excellently played by an top-notch cast including Tcheky Karyo, Michael Wincott and Sigourney Weaver as the Queen of Spain. Interestingly the script shows Columbus mercenary side
       by detailing the deals he wanted in return for discovering this new world before eventually having to return home in disgrace.
     </review>
     <stars>GERARD DEPARDIEU, ARMAND ASSANTE</stars>
     <title>1492 - CONQUEST OF PARADISE</title>
     <year>1992</year>
   </film>
  ▶<film>...</film>
  ▶<film>...</film>
  ▼<film>
     <director>JIM WYNORSKI</director>
     <id>10004</id>
    ▼<review>
       Sequel that still finds the ultimate telephone line, the Devils equivalent of an 0898 number, acquiring victims. The head of a high school taps into the line, hoping to receive supernatural powers to help him in his obsessive lust for a young student. Any
       one who gets in his way suffers a horrible death. With a supporting role for Brigitte Nielsen as a seductress in league with Satan himself.
     </review>
     <stars>PATRICK OBRYAN, DEBBIE JAMES</stars>
     <title>976-EVIL II: THE RETURN</title>
     <year>1988</year>
   </film>
  ▶<film>...</film>
  ▼<film>
     <director>ROBERT DE NIRO</director>
     <id>10006</id>
    ▼<review>
       De Niros directorial debut, taken from co-star Palminteris own stage play. De Niro leads a cast of virtual unknown as a hard-working Bronx bus driver, alarmed at his sons fascination with the local underworld figures. But the lad is involved even deeper
       than he knows and soon hell have to confront the gangsters in an effort to win back his sons respect.
     </review>
     <stars>ROBERT DE NIRO, CHAZZ PALMINTERI</stars>
     <title>A BRONX TALE</title>
     <year>1993</year>
   </film>
  ▶<film>...</film>
  ▼<film>
     <director>ROB REINER</director>
     <id>10008</id>
    ▼<review>
       After the mysterious death of a young Marine in the barracks, a young navy Lawyer (Cruise) is called in to investigate. The further he delves into the mystery the more he is thwarted by the fearsome and unyielding Colonel Nathan Jessup (Nicholson), a man
       who knows the truth and is determined to conceal it....
     </review>
     <stars>TOM CRUISE, JACK NICHOLSON</stars>
     <title>A FEW GOOD MEN</title>
     <year>1992</year>
   </film>
  ▼<film>
     <director>PENNY MARSHALL</director>
     <id>10009</id>
    ▼<review>
       Loosely based on the true story of the all-womens baseball league which was formed during the war while all the male players were fighting overseas. Concentrating on one team, the Rockford Peaches whose members include Davis, Madonna, and Lori Petty.
       Broken-down coach Tom Hanks has to knock them into some sort of shape.
     </review>
     <stars>GEENA DAVIS, TOM HANKS</stars>
     <title>A LEAGUE OF THEIR OWN</title>
     <year>1992</year>
   </film>
  ▼<film>
```

Picture 5. Proof of API using "getAllFilms()" method

Picture 6. shows the servlet that was called in Picture 5. and is responsible for getting a list of all films from the FilmDAO.java class and the format, from the http request, in which the results are to be displayed, string, xml or json format. Upon receiving the information, it forwards the format, the result, meaning list of films, the request and the response to the View and then the View in the MVC takes care of formatting, displaying the results and responding.

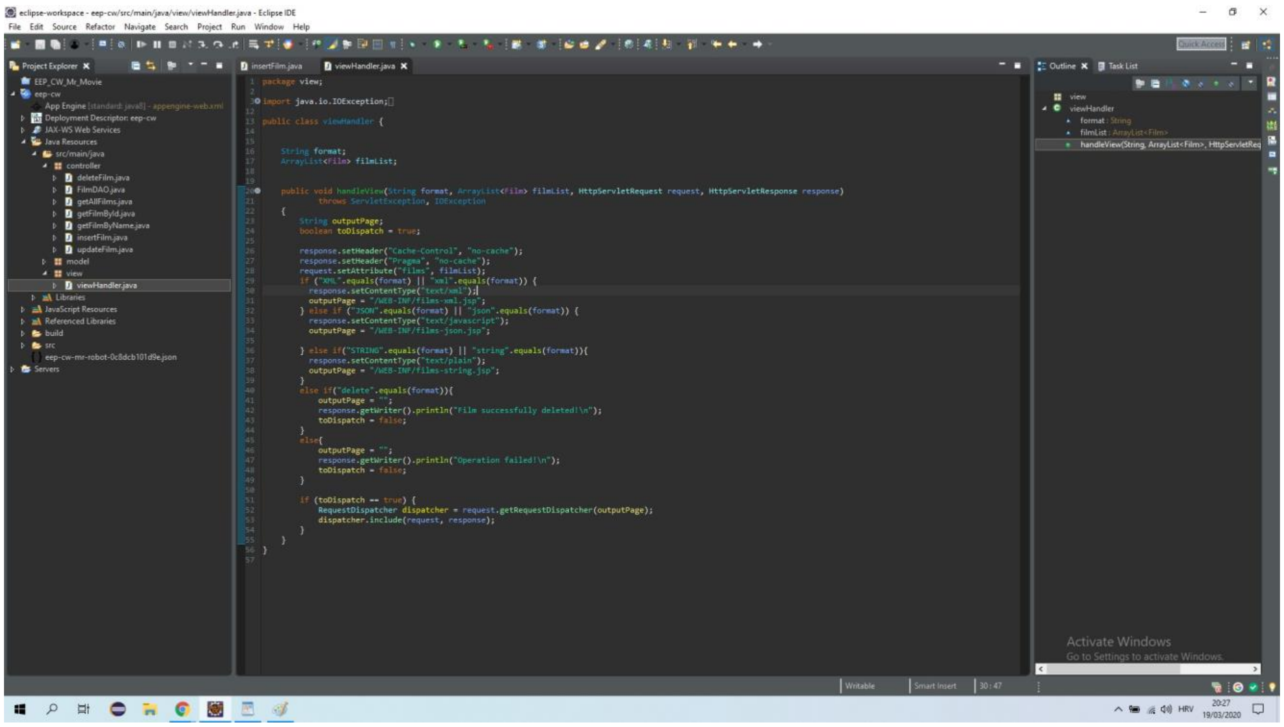


```java
package controller;

import java.io.IOException;

@WebServlet("/getAllFilms")
public class getAllFilms extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public getAllFilms() {
        super();
    }

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

        FilmDAO dao = new FilmDAO();
        ArrayList<Film> filmList ;

        String format = request.getParameter("getAllFilms_format");
        System.out.println("Format: " + format);
        filmList = dao.getAllFilms();


        viewHandler viewHandler = new viewHandler();
        viewHandler.handleView(format, filmList, request, response);
    }

    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
        doGet(request, response);
    }
}
```

```
Mar 19, 2020 9:08:00 PM com.google.appengine.tools.development.AbstractModule startup
INFO: Module instance default is running at http://localhost:8080/
Mar 19, 2020 9:08:00 PM com.google.appengine.tools.development.AbstractModule startup
INFO: The admin console is running at http://localhost:8080/_ah/admin
Mar 19, 2020 9:08:00 PM com.google.appengine.tools.development.DevAppServerImpl doStart
INFO: Dev App Server is now running
Format: STRING
select * from films[10001] 1492 - CONQUEST OF PARADISE (1992) --- Directed by: RIDLEY SCOTT, Staring: GERARD DEPARDIEU, ARMAND ASSANTE<br></br>The story of Columbus discovering America, beautifully shot in Costa Rica and e
[10002] 187 (1997) --- Directed by: KEVIN REYNOLDS, Staring: SAMUEL L.JACKSON, JOHN HEARD<br></br>The title number is the police code for a homicide, and its heard more than once in this tough ghetto school drama. Samuel L
[10003] 9 1/2 WEEKS (1986) --- Directed by: ADRIAN LYNE, Staring: MICKEY ROURKE, KIM BASINGER<br></br>Director Lyne shows his pop video roots in this erotic tale reputedly based on a true story. Basinger falls for stubble-
[10004] 976-EVIL II: THE RETURN (1988) --- Directed by: JIM WYNORSKI, Staring: PATRICK OBRYAN, DEBBIE JAMES<br></br>Sequel that still finds the ultimate telephone line, the Devils equivalent of an 0898 number, acquiring vi
[10005] A BETTER TOMORROW (1986) --- Directed by: JOHN WOO, Staring: CHOW YUN-FAT, LESLIE CHEUNG<br></br>The film which brought international stardom to star Chow Yun-Fat and director John Woo. A roller-coaster of a ride w
```

Picture 6. "getAllFilms()" method implementation

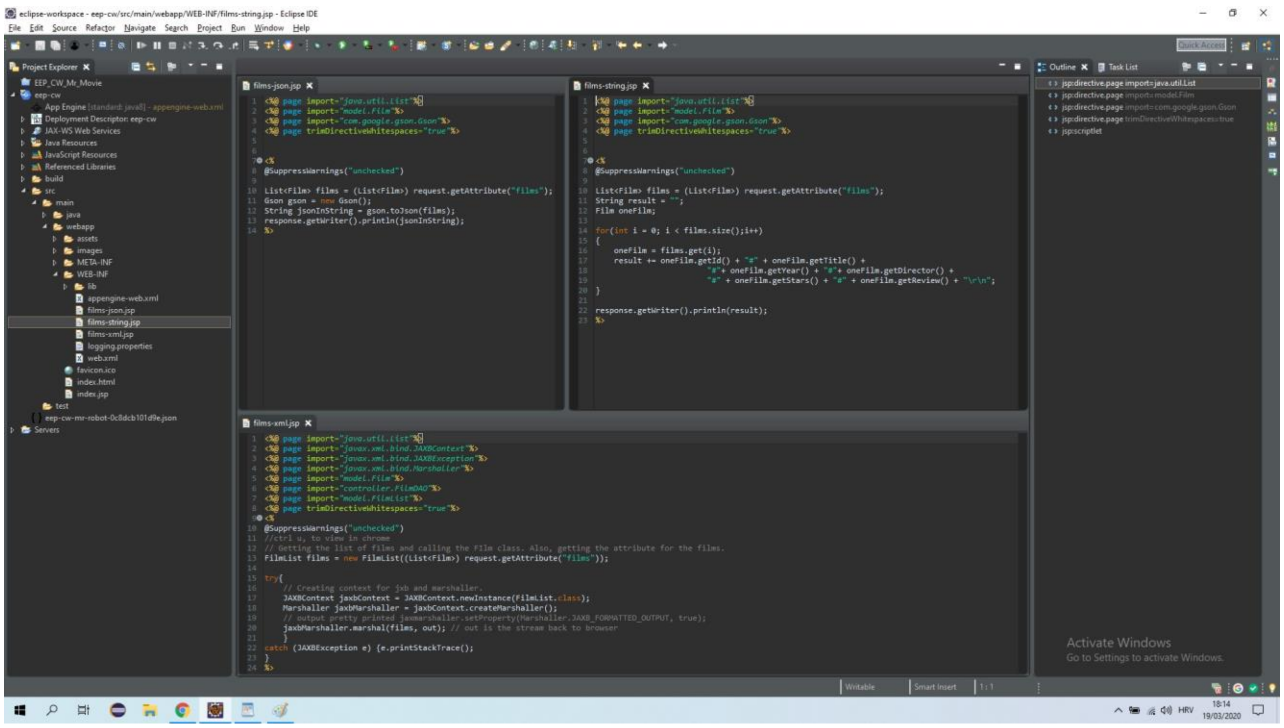
Picture 7. shows the code for the View in MVC and is called upon by each servlet when it needs to display results.



```java
package view;

import java.io.IOException;

public class viewHandler {


    String format;
    ArrayList<Film> filmList;


    public void handleView(String format, ArrayList<Film> filmList, HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException
    {
        String outputPage;
        boolean toDispatch = true;

        response.setHeader("Cache-Control", "no-cache");
        response.setHeader("Pragma", "no-cache");
        request.setAttribute("films", filmList);
        if ("XML".equals(format) || "xml".equals(format)) {
          response.setContentType("text/xml");
          outputPage = "/WEB-INF/films-xml.jsp";
        } else if ("JSON".equals(format) || "json".equals(format)) {
          response.setContentType("text/javascript");
          outputPage = "/WEB-INF/films-json.jsp";

        } else if("STRING".equals(format) || "string".equals(format)){
          response.setContentType("text/plain");
          outputPage = "/WEB-INF/films-string.jsp";
        }
        else if("delete".equals(format)){
            outputPage = "";
            response.getWriter().println("Film successfully deleted!\n");
            toDispatch = false;
        }
        else{
            outputPage = "";
            response.getWriter().println("Operation failed!\n");
            toDispatch = false;
        }

        if (toDispatch == true) {
            RequestDispatcher dispatcher = request.getRequestDispatcher(outputPage);
            dispatcher.include(request, response);
        }
    }
}
```

Picture 7. Implementation of the View in the MVC

Picture 8. Shows proof that the View uses three different .jsp files in order to format the output into each format, a string, an xml or a json format. After formatting the response is then sent.


films-json.jsp

```
<%@ page import="java.util.List"%>
<%@ page import="model.Film"%>
<%@ page import="com.google.gson.Gson"%>
<%@ page trimDirectiveWhitespaces="true"%>


<%
@SuppressWarnings("unchecked")

List<Film> films = (List<Film>) request.getAttribute("films");
Gson gson = new Gson();
String jsonInString = gson.toJson(films);
response.getWriter().println(jsonInString);
%>
```

films-string.jsp

```
<%@ page import="java.util.List"%>
<%@ page import="model.Film"%>
<%@ page import="com.google.gson.Gson"%>
<%@ page trimDirectiveWhitespaces="true"%>


<%
@SuppressWarnings("unchecked")

List<Film> films = (List<Film>) request.getAttribute("films");
String result = "";
Film oneFilm;

for(int i = 0; i < films.size();i++)
{
    oneFilm = films.get(i);
    result += oneFilm.getId() + "#" + oneFilm.getTitle() +
                        "#"+ oneFilm.getYear() + "#"+ oneFilm.getDirector() +
                        "#" + oneFilm.getStars() + "#" + oneFilm.getReview() + "\r\n";
}

response.getWriter().println(result);
%>
```

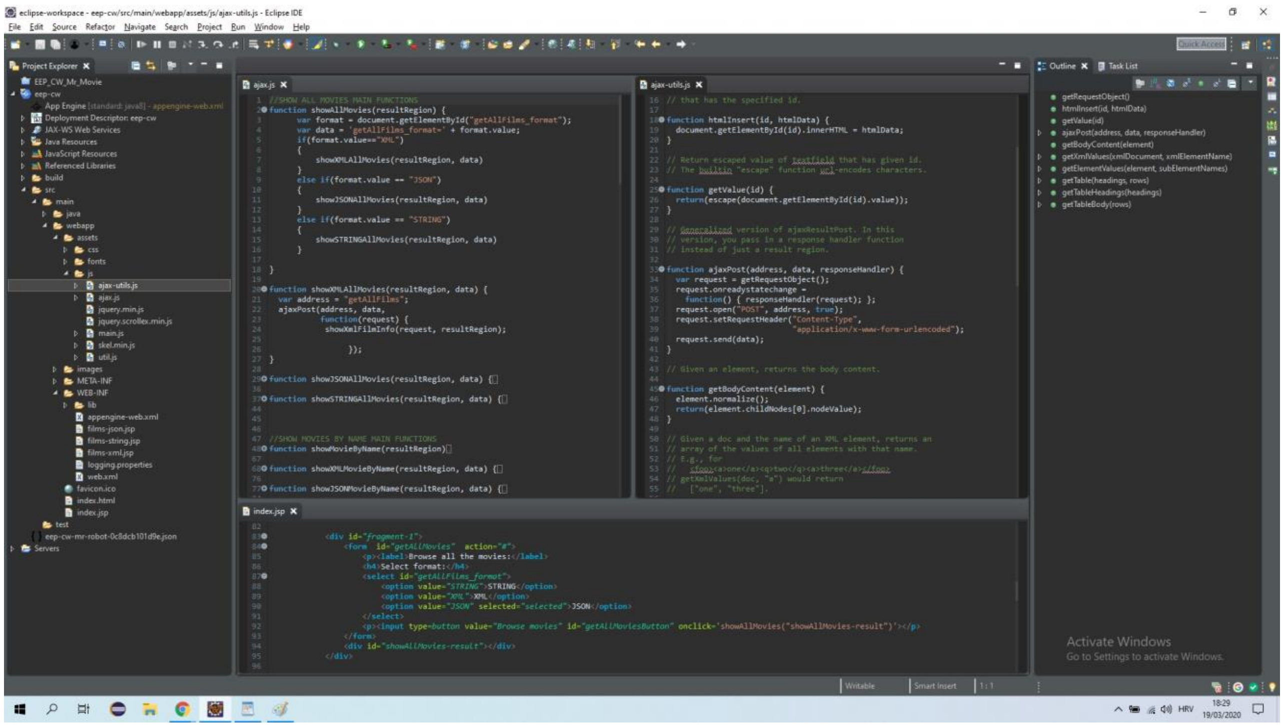
films-xml.jsp

```
<%@ page import="java.util.List"%>
<%@ page import="javax.xml.bind.JAXBContext"%>
<%@ page import="javax.xml.bind.JAXBException"%>
<%@ page import="javax.xml.bind.Marshaller"%>
<%@ page import="model.Film"%>
<%@ page import="controller.FilmDAO"%>
<%@ page import="model.FilmList"%>
<%@ page trimDirectiveWhitespaces="true"%>
<%
@SuppressWarnings("unchecked")
//ctrl u, to view in chrome
// Getting the list of films and calling the FIlm class. Also, getting the attribute for the films.
FilmList films = new FilmList((List<Film>) request.getAttribute("films"));

try{
    // Creating context for jxb and marshaller.
    JAXBContext jaxbContext = JAXBContext.newInstance(FilmList.class);
    Marshaller jaxbMarshaller = jaxbContext.createMarshaller();
    // output pretty printed jaxmarshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
    jaxbMarshaller.marshal(films, out); // out is the stream back to browser
    }
catch (JAXBException e) {e.printStackTrace();
}
%>
```

Picture 8. Three .jps files used for formatting the output

Browser side: Ajax and jQuery

In this section I will go over the main touching points concerning the client or the browser side of the application. Picture 9. shows the html snippet of the index.jsp (homepage of the application) at the bottom that contains html code for invoking the "getAllFilms()" method. Furthermore, it shows the highest abstraction level code of ajax on the left-hand side and the usage of jQuery functions on the right-hand side to generate a POST request to the server.



```
//SHOW ALL MOVIES MAIN FUNCTIONS
function showAllMovies(resultRegion) {
        var format = document.getElementById("getAllFilms_format");
        var data = 'getAllFilms_format=' + format.value;
        if(format.value=="XML")
        {
            showXMLAllMovies(resultRegion, data)
        }
        else if(format.value == "JSON")
        {
            showJSONAllMovies(resultRegion, data)
        }
        else if(format.value == "STRING")
        {
            showSTRINGAllMovies(resultRegion, data)
        }

}

function showXMLAllMovies(resultRegion, data) {
  var address = "getAllFilms";
  ajaxPost(address, data,
           function(request) {
            showXmlFilmInfo(request, resultRegion);

                });
}

function showJSONAllMovies(resultRegion, data) {

function showSTRINGAllMovies(resultRegion, data) {



//SHOW MOVIES BY NAME MAIN FUNCTIONS
function showMovieByName(resultRegion)

function showXMLMovieByName(resultRegion, data) {

function showJSONMovieByName(resultRegion, data) {
```

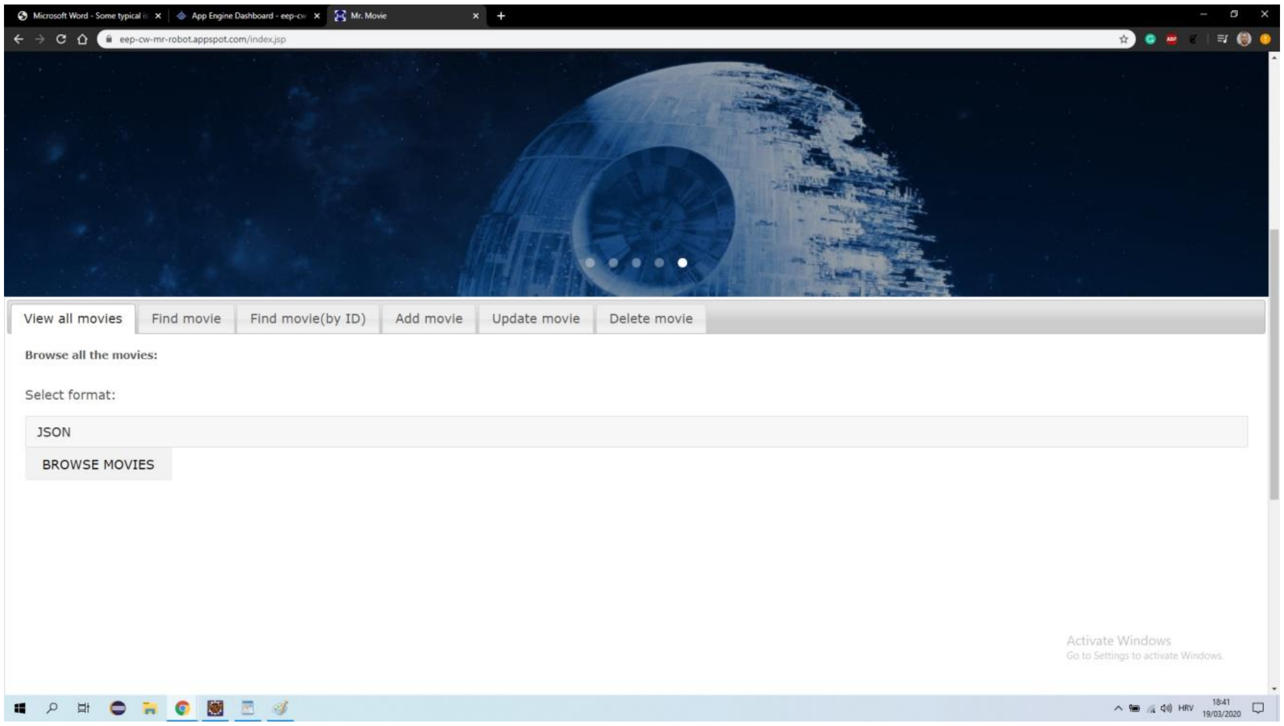
```
// that has the specified id.

function htmlInsert(id, htmlData) {
  document.getElementById(id).innerHTML = htmlData;
}

// Return escaped value of textfield that has given id.
// The builtin "escape" function url-encodes characters.

function getValue(id) {
  return(escape(document.getElementById(id).value));
}

// Generalized version of ajaxResultPost. In this
// version, you pass in a response handler function
// instead of just a result region.

function ajaxPost(address, data, responseHandler) {
  var request = getRequestObject();
  request.onreadystatechange =
    function() { responseHandler(request); };
  request.open("POST", address, true);
  request.setRequestHeader("Content-Type",
                           "application/x-www-form-urlencoded");
  request.send(data);
}

// Given an element, returns the body content.

function getBodyContent(element) {
  element.normalize();
  return(element.childNodes[0].nodeValue);
}

// Given a doc and the name of an XML element, returns an
// array of the values of all elements with that name.
// E.g., for
//   <foo><a>one</a><q>two</q><a>three</a></foo>
// getXmlValues(doc, "a") would return
//   ["one", "three"].
```

```
            <div id="fragment-1">
                <form  id="getAllMovies"  action="#">
                    <p><label>Browse all the movies:</label>
                    <h4>Select format:</h4>
                    <select id="getAllFilms_format">
                        <option value="STRING">STRING</option>
                        <option value="XML">XML</option>
                        <option value="JSON" selected="selected">JSON</option>
                    </select>
                    <p><input type=button value="Browse movies" id="getAllMoviesButton" onclick='showAllMovies("showAllMovies-result")'></p>
                </form>
                <div id="showAllMovies-result"></div>
            </div>
```

eclipse-workspace - eep-cw/src/main/webapp/assets/js/ajax-utils.js - Eclipse IDE
ajax.js
ajax-utils.js
index.jsp
Outline
getRequestObject()
htmlInsert(id, htmlData)
getValue(id)
ajaxPost(address, data, responseHandler)
getBodyContent(element)
getXmlValues(xmlDocument, xmlElementName)
getElementValues(element, subElementNames)
getTable(headings, rows)
getTableHeadings(headings)
getTableBody(rows)
Writable
Smart Insert
1 : 1

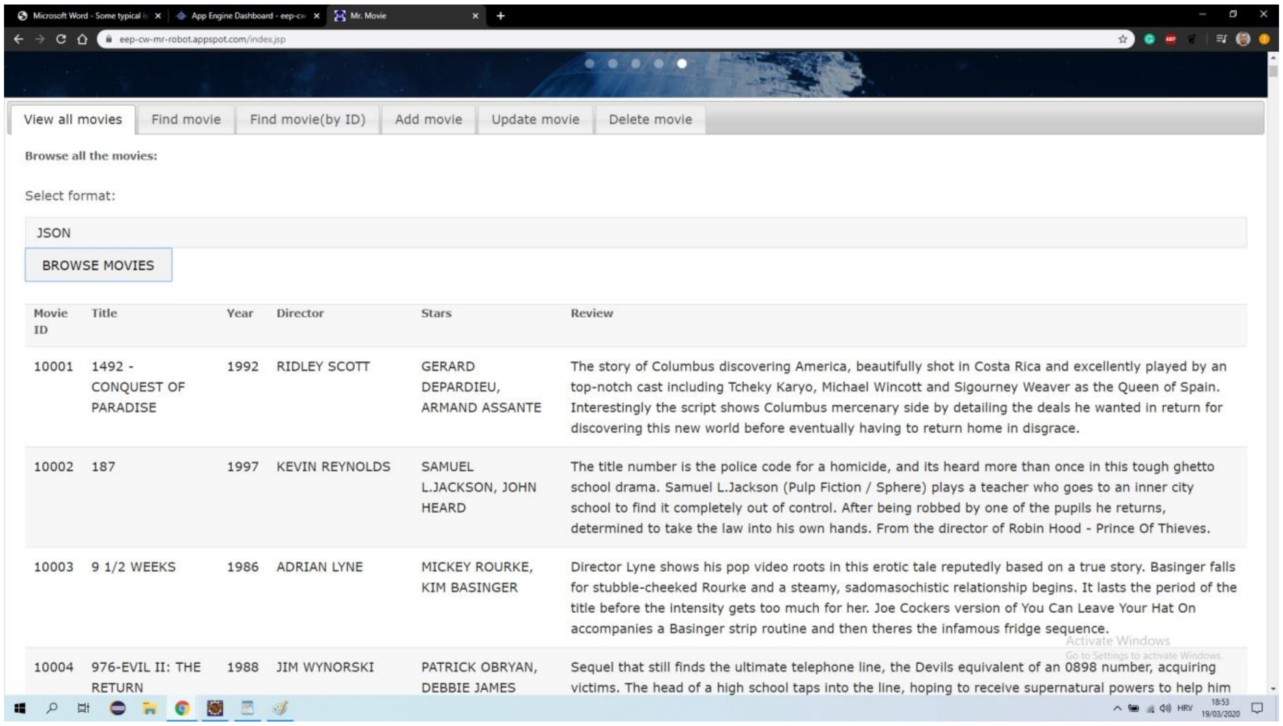
Picture 9. HTML, Ajax and jQuery snippets for invoking “getAllFilm()” method

Picture 10. is a visual representation of Picture 9. and proves that jQuery User Interface was used in the front-end of the application. Not only that, as you can see in the url of the browser, it is not based on a local server. Which means that the application is cloud based and uses Google Cloud Platform. However, the database is not stored on the cloud but rather at the MMU database server, even though I have put in the effort to make it possible.



Picture 10. Front-end, jQuery UI for “getAllFilms()”

Picture 11. Shows the result of the invoked method “getAllFilms()” using json format, and thus proving that the format is successfully passed inside a request to a servlet i.e. Controller which accesses the Model and retrieves the data from the database which then forwards to the Viewer. The Viewer then formats the data and sends it back to the client where ajax and jQuery shape the data into a table and display it as seen in the Picture 11.



Browse all the movies:

Select format:

JSON

Movie ID	Title	Year	Director	Stars	Review
10001	1492 - CONQUEST OF PARADISE	1992	RIDLEY SCOTT	GERARD DEPARDIEU, ARMAND ASSANTE	The story of Columbus discovering America, beautifully shot in Costa Rica and excellently played by an top-notch cast including Tcheky Karyo, Michael Wincott and Sigourney Weaver as the Queen of Spain. Interestingly the script shows Columbus mercenary side by detailing the deals he wanted in return for discovering this new world before eventually having to return home in disgrace.
10002	187	1997	KEVIN REYNOLDS	SAMUEL L.JACKSON, JOHN HEARD	The title number is the police code for a homicide, and its heard more than once in this tough ghetto school drama. Samuel L.Jackson (Pulp Fiction / Sphere) plays a teacher who goes to an inner city school to find it completely out of control. After being robbed by one of the pupils he returns, determined to take the law into his own hands. From the director of Robin Hood - Prince Of Thieves.
10003	9 1/2 WEEKS	1986	ADRIAN LYNE	MICKEY ROURKE, KIM BASINGER	Director Lyne shows his pop video roots in this erotic tale reputedly based on a true story. Basinger falls for stubble-cheeked Rourke and a steamy, sadomasochistic relationship begins. It lasts the period of the title before the intensity gets too much for her. Joe Cockers version of You Can Leave Your Hat On accompanies a Basinger strip routine and then theres the infamous fridge sequence.
10004	976-EVIL II: THE RETURN	1988	JIM WYNORSKI	PATRICK OBRYAN, DEBBIE JAMES	Sequel that still finds the ultimate telephone line, the Devils equivalent of an 0898 number, acquiring victims. The head of a high school taps into the line, hoping to receive supernatural powers to help him

Picture 11. Results of “getAllFilms()” invoked in the browser