Assignment 2

Java Enterprise

Movie Feedback

991667681

Balwinder Kaur

Step 1

As usual we need to add a new project , click on the new than other than starter project

A screenshot of a computer

Description automatically generated  
  
A screenshot of a computer

Description automatically generated  
  
  
Than we need to add only those dependencies that are required to make this program and only those that we have covered till this date like h2, Lombok , spring dev tools , jdbc ,spring web ,thymleaf

A screenshot of a computer

Description automatically generated  
  
  
So according to the instructions given in the assignment we need to make a POJO Class in which we need to declare some Variables like movie name, review and is as well. Than we gonna some Lombok annotations to make this class a little bit different

In case of beans package I added the name as Feedback which like that of Review

A screenshot of a computer

Description automatically generated  
  
  
Now I am declaring the required variables in this class

String Moviename ,String review ,Long id, LocalDatetime

And than I am adding the required Lombok Annotations @Data ,@AllArgsConstructor and @NoArgsConstructor just to add the getter and setters, toString() and the hashCode(). Here is the required screenshot below

A screenshot of a computer

Description automatically generated with medium confidence  
  
  
Than after that I making schema.sql and data.sql file in the templates directory of the src/main

A screenshot of a computer

Description automatically generated  
  
Than data.sql file

A screenshot of a computer

Description automatically generated  
  
  
  
Than in schema.sql file I am making a new table called feedback as shown below

**CREATE** **TABLE** Feedback (

id BIGINT **PRIMARY** **KEY** AUTO\_INCREMENT,

Moviename **VARCHAR**(255),

review TEXT,

datetime **TIMESTAMP**

);

A screenshot of a computer

Description automatically generated with medium confidence  
  
  
  
  
Than I am inserting some entries in this new table that we need to do in the data.sql file

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('KaranArjun' ,'Great Emotional movie', '2023-06-16 06:12:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('Inception' ,'Best Science Fiction movie', '2023-06-16 12:30:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('Border' ,'Best indian movie of all time', '2023-06-16 11:12:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('JohnCarter' ,'Explainning gravity in this movie', '2023-06-16 05:12:00');

A screenshot of a computer

Description automatically generated with medium confidence  
  
  
  
  
Now I am enabling the h2-console orthe virtual database in the application.properties

A screenshot of a computer

Description automatically generated with medium confidence  
  
Now coming back to src/main/java we need to make the controllers package declare some methods that will get mapp to the index page as shown

A screenshot of a computer

Description automatically generated with medium confidence  
  
Than we need to make index.html page where we are going to make a form and table so that controller class can handle the post mapping request

A screenshot of a computer

Description automatically generated  
  
The most thing in the index.html page is to add

A picture containing text, font, line, screenshot

Description automatically generated  
  
Than I made a unorderd list where I am adding only one list items that will iterate over the Feedback table that I made in the schema.sql file via using thymleaf dependencies as shown in the figure below.

Used italics to show the name of the moviename ,Review ,Present date and time

A picture containing text, screenshot, font

Description automatically generated  
  
  
  
Than we added a form here where we are adding the moviename , review and datetime via using thymleaf that th:field th:value that will assign value of date and time as shown in the figure below

A screenshot of a computer

Description automatically generated with medium confidence  
  
Now come back to the Controller class and now declare a method with @PostMapping(“/”)to the form

And in that method we are model argument and calling the model attribute with the thymleaf object that is in the th:object =${newFeedback} and we are making this object to that of the object of the Feedback class so that we can set the new values of the details submitted by the user in the form

A screenshot of a computer

Description automatically generated with medium confidence  
  
  
  
  
Also not forgot to addattribute in the index method as well so whenever the index page is returned it should show the new review added as well

A picture containing text, screenshot, font, line

Description automatically generated  
  
  
  
Now let’s go to the database package as make a database class

A screenshot of a computer

Description automatically generated  
  
First Adding the repository annotation on the top of the class and now we declare a method that will

A screenshot of a computer

Description automatically generated   
  
A blurry image of a movie theater

Description automatically generated with low confidence  
  
  
  
A blurry image of a red object

Description automatically generated with low confidence  
  
  
  
  
A screenshot of a movie

Description automatically generated with medium confidence  
  
  
  
A screen shot of a movie

Description automatically generated with low confidence  
  
  
  
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Beans Package  
package ca.sheridancollege.kohliman.beans;

import java.time.LocalDateTime;

import lombok.AllArgsConstructor;

import lombok.Data;

import lombok.NoArgsConstructor;

@Data

@AllArgsConstructor

@NoArgsConstructor

public class Feedback {

public Long id;

public String Moviename;

public String review;

public LocalDateTime datetime;

}  
  
  
Controller package

package ca.sheridancollege.kohliman.controllers;

import java.time.LocalDateTime;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.ModelAttribute;

import org.springframework.web.bind.annotation.PostMapping;

import ca.sheridancollege.kohliman.beans.Feedback;

import ca.sheridancollege.kohliman.database.DatabaseAccess;

@Controller

public class FeedbackController {

@Autowired

private DatabaseAccess dta;

public FeedbackController(DatabaseAccess dta) {

this.dta = dta;

}

@GetMapping("/")

public String index(Model model) {

model.addAttribute("newFeedback", new Feedback());

model.addAttribute("Feedback", dta.everyfeedback());

return "index";

}

@PostMapping("/")

public String newFeedback(Model model ,@ModelAttribute("newFeedback") Feedback fd) {

fd.setDatetime((LocalDateTime.now()));

model.addAttribute("newFeedback",new Feedback());

dta.addingfeedback(fd);

return"redirect:/";

}

}  
  
  
Database class

package ca.sheridancollege.kohliman.database;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.jdbc.core.BeanPropertyRowMapper;

import org.springframework.jdbc.core.namedparam.MapSqlParameterSource;

import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;

import org.springframework.stereotype.Repository;

import ca.sheridancollege.kohliman.beans.Feedback;

@Repository

public class DatabaseAccess {

@Autowired

private NamedParameterJdbcTemplate jdbc;

public DatabaseAccess(NamedParameterJdbcTemplate jdbc) {

this.jdbc = jdbc;

}

public void addingfeedback(Feedback fd) {

MapSqlParameterSource namedParameters = new MapSqlParameterSource();

namedParameters.addValue("Moviename", fd.getMoviename());

namedParameters.addValue("review", fd.getReview());

namedParameters.addValue("datetime", fd.getDatetime());

String query = "INSERT INTO Feedback (Moviename, review, datetime) VALUES (:Moviename, :review, :datetime)";

int rowsAffected = jdbc.update(query, namedParameters);

if (rowsAffected > 0)

System.out.println("Review inserted");

}

public List<Feedback> everyfeedback() {

MapSqlParameterSource namedParameters = new MapSqlParameterSource();

String query = "SELECT \* FROM Feedback";

return jdbc.query(query, namedParameters, new BeanPropertyRowMapper<>(Feedback.class));

}

}  
  
  
  
  
index.html

<!DOCTYPE html>

<html lang=*"en"* xmlns:th=*"http://www.thymeleaf.org"*>

<head>

<meta charset=*"UTF-8"*>

<title>Movie Feedback</title>

<style>

**x** **body** {

font-family: *Arial, sans-serif*;

margin: *0*;

padding: *20px*;

background-color: *lightpink*;

color: *white*; /\* Set text color to white \*/

}

**h1** {

color: *white*;

text-align: *center*;

margin-bottom: *30px*;

}

**ol** {

list-style-type: *none*;

padding: *0*;

}

**li** {

/\* Remove the white boxes \*/

background-color: *transparent*;

border-radius: *0*;

padding: *0*;

margin-bottom: *20px*;

box-shadow: *none*;

}

**li***:last-child* {

margin-bottom: *0*;

}

**h2** {

color: *white*;

margin-top: *0*;

font-weight: *normal*;

font-size:*30px*;

}

**i** {

font-style: *italic*;

color: *#888*;

}

**span** {

font-weight: *bold*;

color: *white*;

}

**form** {

background-color: *#fff*;

border-radius: *5px*;

padding: *20px*;

box-shadow: *0px 2px 5px rgba(0, 0, 0, 0.1)*;

}

**label** {

font-weight: *bold*;

display: *block*;

margin-bottom: *10px*;

background-color: *lightpink*;

padding: *5px*;

}

**input**[type="text"]**,**

**textarea** {

width: *100%*;

padding: *10px*;

border: *1px solid #ccc*;

border-radius: *4px*;

resize: *vertical*;

}

**button**[type="submit"] {

background-color: *#ff5a5f*;

color: *#fff*;

border: *none*;

padding: *10px 20px*;

border-radius: *4px*;

cursor: *pointer*;

}

**button**[type="submit"]*:hover* {

background-color: *#ff4045*;

}

/\* Background Image Styles \*/

*.background-image* {

position: *fixed*;

top: *0*;

left: *0*;

z-index: *-1*;

width: *100%*;

height: *100%*;

filter: *blur(15px)*;

}

</style>

</head>

<body>

<h1>Movie Feedback</h1>

<hr>

<h2>Reviews</h2>

<hr>

<div class=*"background-image"*>

<img src=*"../static/img/m.jpeg"* th:src=*"@{/img/m.jpeg}"* style="width: *100%*; height: *100%*;" />

</div>

<ol>

<li th:each=*"f : ${Feedback}"*>

<h2><i>Movie Name:</i> <span th:text=*"${f.Moviename}"*>Movie name</span></h2>

<h2><i>Review:</i> <span th:text=*"${f.review}"*>Review</span></h2>

<h2><i>Present Date and Time:</i> <span th:text=*"${#temporals.format(f.datetime, 'dd/mm/yyyy HH:mm')}"*>Date and time</span></h2>

</li>

</ol>

<h1>Add your Review/Feedback</h1>

<form th:action=*"@{/}"* method=*"post"* th:object=*"${newFeedback}"*>

<br>

<label for=*"Moviename"*>Movie Name:</label>

<input type=*"text"* id=*"moviename"* th:field=*"\*{Moviename}"*>

<br>

<br>

<label for=*"review"*>Review:</label>

<textarea id=*"review"* th:field=*"\*{review}"*></textarea>

<br>

<input type=*"hidden"* th:field=*"\*{datetime}"* th:value=*"${#temporals.format(#temporals.createNow(), 'yyyy-MM-dd HH:mm:ss')}"*>

<button type=*"submit"*>CLick Here to Submit</button>

</form>

</body>

</html>

data.sql

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('KaranArjun' ,'Great Emotional movie', '2023-06-16 06:12:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('Inception' ,'Best Science Fiction movie', '2023-06-16 12:30:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('Border' ,'Best indian movie of all time', '2023-06-16 11:12:00');

**INSERT** **INTO** Feedback(Moviename, review ,datetime) **VALUES**('JohnCarter' ,'Explainning gravity in this movie', '2023-06-16 05:12:00');

schema.sql

**CREATE** **TABLE** Feedback (

id BIGINT **PRIMARY** **KEY** AUTO\_INCREMENT,

Moviename **VARCHAR**(255),

review TEXT,

datetime **TIMESTAMP**

);

Summary

So in this Assignment we used all the dependencies that we have covered till yet and used them in different classes. Like Lombok we have used in the beans package for get and set operation and thymleaf for dynamic data on the webpage.Than we have used the jdbc template and database as well. Obviously the springbootdev and spring web we using since the start of the semester as they are important in order to map to the web page and also to control the class using @Controller package. So in this we are declaring some variables in the beans package for the movie feedback and in the controller package we declaring the index method to getmapped to the index page and than we are declaring the another method that is used for getting the values from the form that we are using in the index page to get new feedback from the customer that then we are calling a method from the database package addfeedback() that will add to the table that is above the form in the index page and than it will redirect the index page and refreshed page will show the new data is being inserted in that table . This is like we are adding some attributes to the namedParameters variables that is the key values , first the message and getting the values using get method from beans package . At last we are inserting the query in the feedback table where we are adding value :Moviename :review :datetime that are the values of Mapsql parameters where they are dynamically adding values and and if the rows are affected in the database table than a line review is added is printed on the console.Don’t forget the data.sql that will add the queries to the table that will appear initially when we will run the tomcat server . and schema.sql file which will make a table.

Therefore this is a small summary how this works .

Thanks