Review-1

JAVA project

Team: Bit busters

Team member:

Priyanshu navdev

Aarab kumar

Raunak raj

Kashish

* Creating a new project.

We will be using Netbeans IDE for our project.

There we will create a new project with the following steps:

1. Create a new project
2. Select Java with Maven from project type and web application from project category.
3. We will give a project name(E-commerse) and choose its location.
4. Then we choose its server(Apache Tomcat or Tom EE) and Java EE version (Java EE version 8)

* Project structure.

Our project is a web application administrator(owner) can list their product, their prices, category and user(costumer) can it from there.

In this project we will be using:

* HTML, CSS, JAVASCRIPT for web appearance
* JAVA (with Maven) for backend
* MySQL for database
* hibernate for data dependencies and data automation.

Some hibernate dependencies:

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate</artifactId>

<version>3.2.7.ga</version>

<type>jar</type>

</dependency>

<dependency>

<groupId>javax</groupId>

<artifactId>javaee-api</artifactId>

<version>8.0.1</version>

<scope>provided</scope>

</dependency>

Project will have two login system :

1. As administrator (owner): Here they can list their shop and product.
2. As user (costumer) : They can browse product and can buy them.

* Database schemas:

In the database for project, we will create following tables and schemas:

* User table

Userid, username, useremail, usepassword, userphone, useradd,

* Category table

Categoryid, categorytitle, categorydiscripliob

* Product

pid, ptitle, pprice, pdisc, pphoto,pquantity,cid

* Order table

Oid, odate,otime,oproduct, deliveryadd, paymentmethod, pid

* Payment table

Payamount, payid, paymethod, payday,oid

* Creating sql table

In this project we will not write sql query directly in mysql but we will be creating Entities for those table in java language and then we map them with help of hibernate to create tables as follws.

First we create a class for our table then we do following cooding,

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence. Entity;

import javax.persistence. Id;

@Entity

public class User {

@Id

@GeneratedValue(strategy GenerationType. IDENTITY)

@Column(length = 10, name = "user\_id")

private int userId;

@Column(length = 100, name = "user\_name")

private String userName;

@Column(length = 100, name = "user\_email")

private String userEmail;

@Column(length = 100, name = "user\_password")

private String userPassword;

@Column(length = 12, name = "user phone")

private String userPhone;

@Column(length = 1500, name "user\_address")

private String userAddress;

}

//Now we create their constructors

public User(int userId, String userName, String userEmail, String userPassword, String userPhone, String userAddress)

this.userId = userId;

this.userName = userName;

this.userEmail userEmail;

this.userPassword userPassword;

this.userPhone userPhone;

this.userAddress userAddress;

}

* Same with other tables.

//We will a hibernate.cfg.xlm file to create these table in database.

!DOCTYPE hibernate-configuration SYSTEM

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name="connection.driver\_class">com.mysql.jdbc.Driver</property> <property name="connection.url">jdbc:mysql://localhost:3306/mycart</property>

<property name="connection.username">root</property> <property name="connection.password">Ary@n</property>

<property name="dialect">org.hibernate.dialect.MySQL57Dialect</property>

<property name="hbm2ddl.auto">update</property>

<property name="show\_sql">true</property>

<mapping class="com.learn.mycart.entities.User" />

</session-factory>

</hibernate-configuration>

* Database connectivity
* With the help of hibernate and its some dependencies we can connect our database to our program and also can create table in java as entities .

The dependence is :

<!-- https://mvnrepository.com/artifact/com.mysql/mysql-connector-j -->

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<version>8.4.0</version>

</dependency>