

ECサイト初級

購入履歴機能の作成

6 時間目

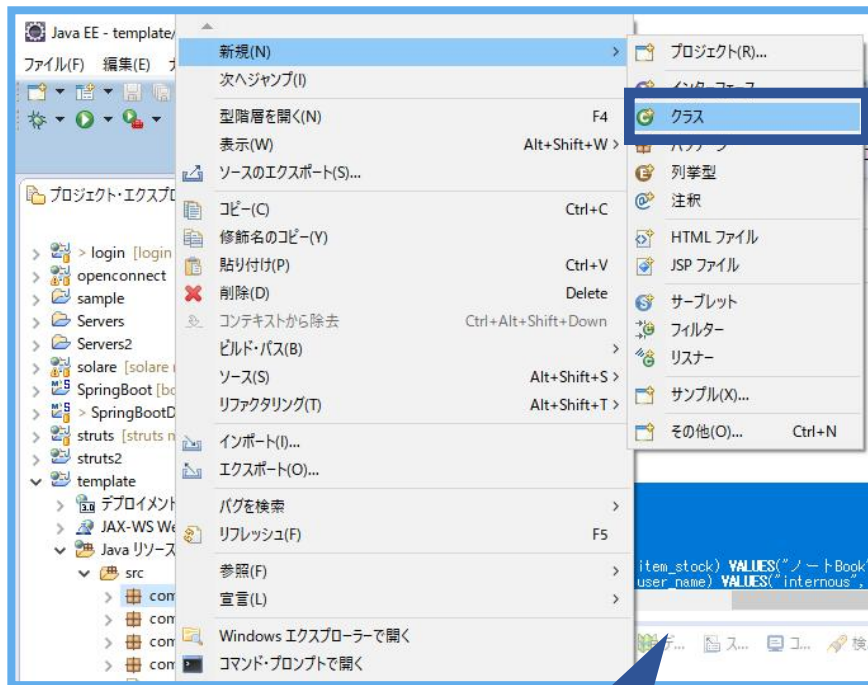
作業目次

作成機能一覧

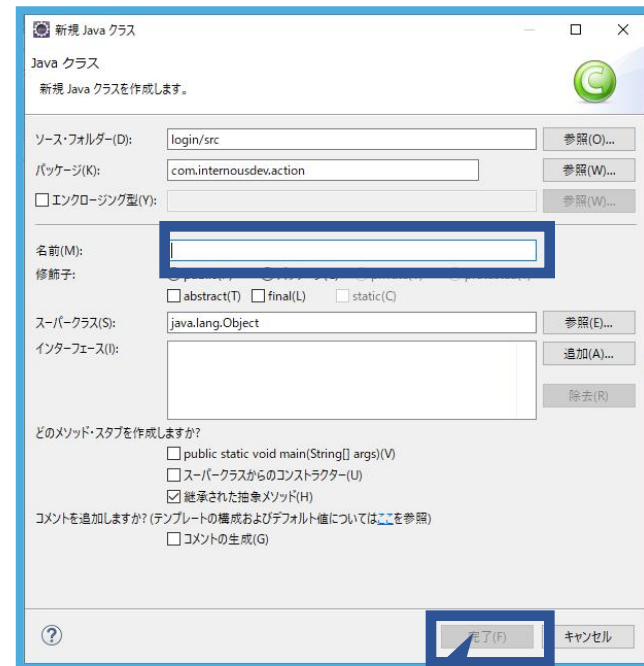
1. ログイン認証機能
2. ユーザー登録機能
3. 商品購入機能
- 4. 購入履歴機能**

MyPageDTO.javaの作成

1 MyPageDTO.java



① 「com.internousdev.template.dto」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「MyPageDTO」を入力し、完了ボタンをクリックします。

MyPageDTO.javaの作成

③ 以下の内容を写経します。

MyPageDTO .java
(javaファイル)

```
package com.internousdev.template.dto;

public class MyPageDTO {

    private String itemName;
    private String totalPrice;
    private String totalCount;
    private String payment;

    public String getItemName() {
        return this.itemName;
    }
    public void setItemName(String itemName) {
        this.itemName = itemName;
    }
}
```

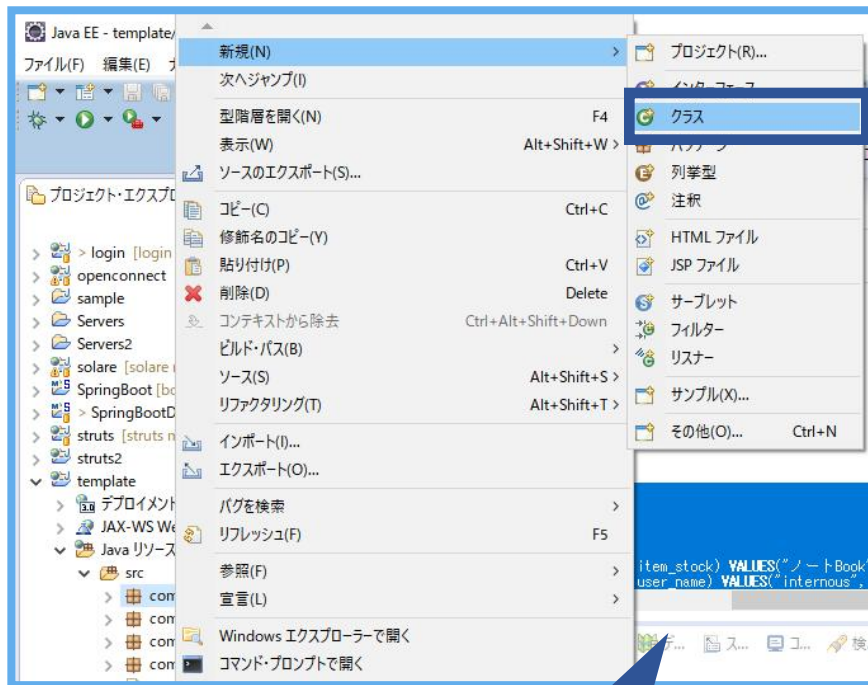
次へ続きます。

前の続きです。

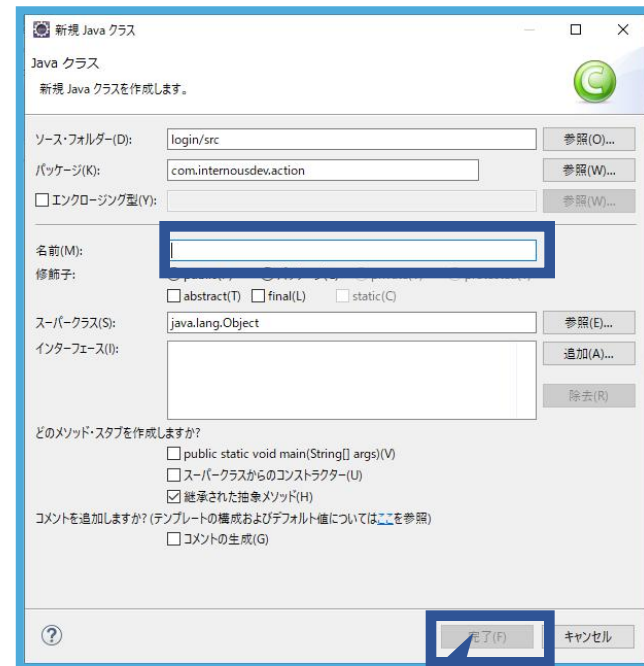
```
public String getTotalPrice() {  
    return this.totalPrice;  
}  
public void setTotalPrice(String totalPrice) {  
    this.totalPrice = totalPrice;  
}  
public String getTotalCount() {  
    return this.totalCount;  
}  
public void setTotalCount(String totalCount) {  
    this.totalCount = totalCount;  
}  
public String getPayment() {  
    return this.payment;  
}  
public void setPayment(String payment) {  
    this.payment = payment;  
}  
}
```

MyPageDAO.javaの作成

2 MyPageDAO.java



① 「com.internousdev.template.dao」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「MyPageDAO」を入力し、完了ボタンをクリックします。

MyPageDAO.javaの作成

③ 以下の内容を写経します。

MyPageDAO .java
(javaファイル)

```
package com.internousdev.template.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import com.internousdev.template.dto.MyPageDTO;
import com.internousdev.template.util.DBConnector;

public class MyPageDAO {

    public MyPageDTO getMyPageUserInfo(String item_transaction_id, String user_master_id) {

        DBConnector db = new DBConnector();
        Connection con = db.getConnection();
        MyPageDTO myPageDTO = new MyPageDTO();
    }
}
```

次へ続きます。

MyPageDAO.javaの作成

テーブル結合をしています。
参考：MySQLコマンドプロンプト

前の続きです。

```
String sql = "SELECT iit.item_name, ubit.total_price, ubit.total_count,  
ubit.pay FROM user_buy_item_transaction ubit LEFT JOIN item_info_transaction iit ON  
ubit.item_transaction_id = iit.id WHERE ubit.item_transaction_id = ? AND  
ubit.user_master_id = ? ORDER BY ubit.insert_date DESC";
```

iitとは：item_info_transaction テーブルを再定義したもの

ubitとは：user_buy_item_transaction テーブルを再定義したもの

```
try {  
    PreparedStatement ps = con.prepareStatement(sql);  
    ps.setString(1, item_transaction_id);  
    ps.setString(2, user_master_id);  
    ResultSet rs = ps.executeQuery();  
  
    if(rs.next()) {  
        myPageDTO.setItemName(rs.getString("item_name"));  
        myPageDTO.setTotalPrice(rs.getString("total_price"));  
        myPageDTO.setTotalCount(rs.getString("total_count"));  
        myPageDTO.setPayment(rs.getString("pay"));  
    }  
}
```

次へ続きます。

MyPageDAO.javaの作成

前の続きです。

```
    } catch(Exception e) {
        e.printStackTrace();
    } finally {
        try {
            con.close();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }
    return myPageDTO;
}

public int buyItemHistoryDelete(String item_transaction_id, String user_master_id) {

    DBConnector db = new DBConnector();
    Connection con = db.getConnection();

    String sql = "DELETE FROM user_buy_item_transaction WHERE
                item_transaction_id = ? AND user_master_id = ?";
    PreparedStatement ps;
    int result = 0;
```

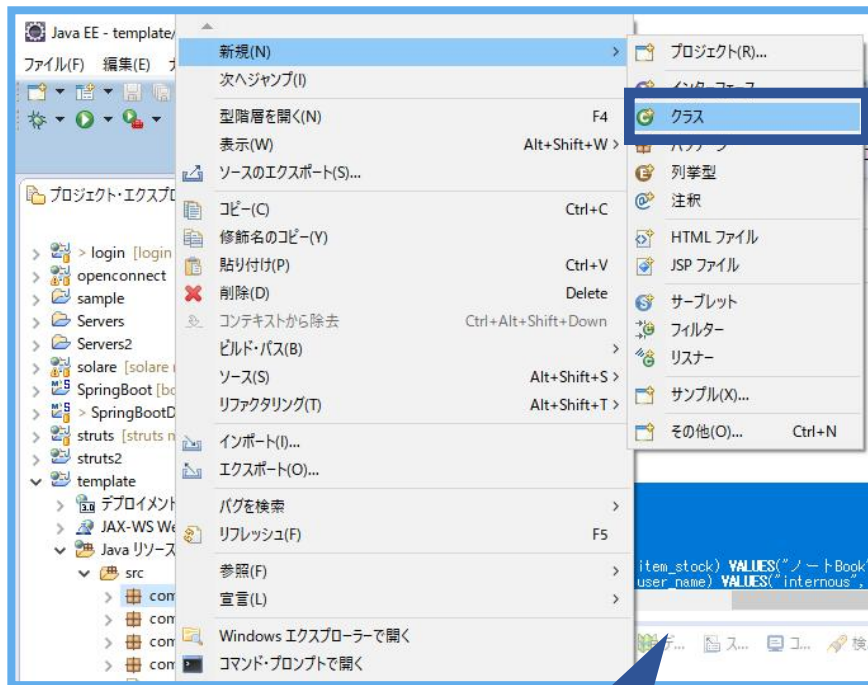
次へ続きます。

前の続きです。

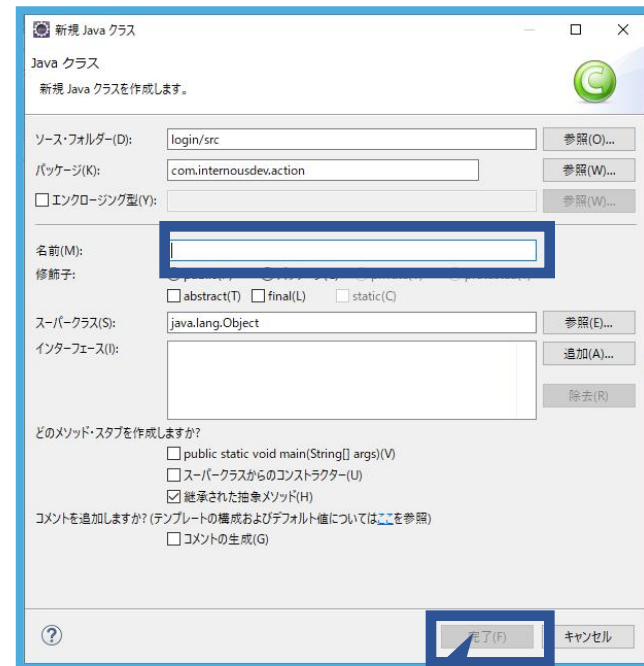
```
try {  
    ps = con.prepareStatement(sql);  
    ps.setString(1, item_transaction_id);  
    ps.setString(2, user_master_id);  
    result = ps.executeUpdate();  
  
} catch(SQLException e) {  
    e.printStackTrace();  
}  
finally {  
    try {  
        con.close();  
    } catch(Exception e) {  
        e.printStackTrace();  
    }  
}  
return result;  
}  
}
```

MyPageAction.javaの作成

3 MyPageAction.java



① 「com.internousdev.template.action」を選択し、「右クリック」「新規」「クラス」を選択します。



② 「名前(M):」欄に「MyPageAction」を入力し、完了ボタンをクリックします。

MyPageAction.javaの作成

③ 以下の内容を写経します。

MyPageAction .java
(javaファイル)

```
package com.internousdev.template.action;
import java.sql.SQLException;
import java.util.Map;
import org.apache.struts2.interceptor.SessionAware;
import com.internousdev.template.dao.MyPageDAO;
import com.internousdev.template.dto.MyPageDTO;
import com.opensymphony.xwork2.ActionSupport;

public class MyPageAction extends ActionSupport implements SessionAware{
    private Map<String, Object> session;
    private String deleteFlg;
    private String result;
    public String execute(){
        MyPageDAO myPageDAO = new MyPageDAO();
        MyPageDTO myPageDTO = new MyPageDTO();

        // 商品履歴を削除しない場合
        if(deleteFlg == null) {
```

次へ続きます。

MyPageAction.javaの作成

前の続きです。

```
String item_transaction_id = session.get("id").toString();
String user_master_id = session.get("login_user_id").toString();

myPageDTO = myPageDAO.getMyPageUserInfo(item_transaction_id,
user_master_id);

session.put("buyItem_name", myPageDTO.getItemName());
session.put("total_price", myPageDTO.getTotalPrice());
session.put("total_count", myPageDTO.getTotalCount());
session.put("total_payment", myPageDTO.getPayment());
session.put("message", "");
// 商品履歴を削除する場合
} else if(deleteFlg.equals("1")) {
    delete();
}
result = SUCCESS;
return result;
}
```

次へ続きます。

MyPageAction.javaの作成

前の続きです。

```
public void delete(){  
  
    MyPageDAO myPageDAO = new MyPageDAO();  
  
    String item_transaction_id = session.get("id").toString();  
    String user_master_id = session.get("login_user_id").toString();  
  
    int res = myPageDAO.buyItemHistoryDelete(item_transaction_id,  
user_master_id);  
  
    if(res > 0) {  
        session.put("message", "商品情報を正しく削除しました。");  
    } else if(res == 0) {  
        session.put("message", "商品情報の削除に失敗しました。");  
    }  
}
```

次へ続きます。

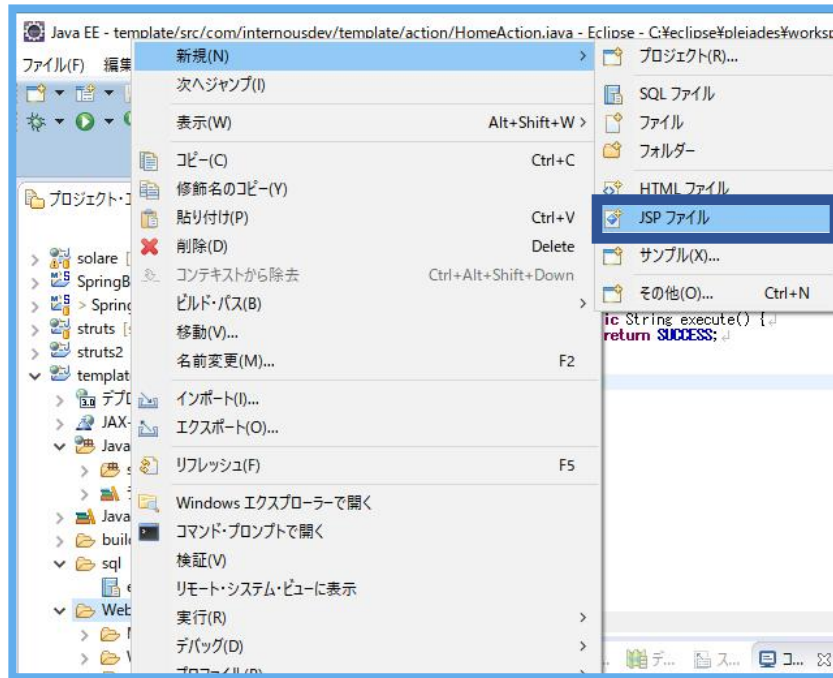
MyPageAction.javaの作成

前の続きです。

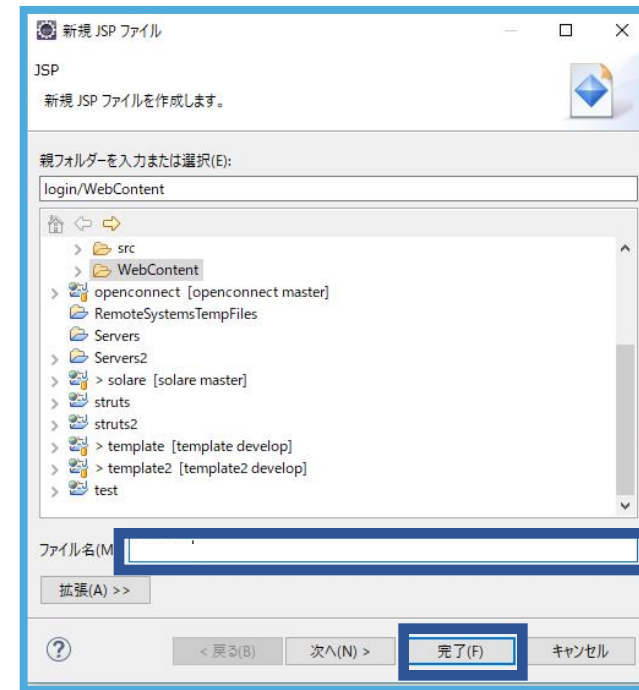
```
public String getDeleteFlg() {  
    return deleteFlg;  
}  
  
public void setDeleteFlg(String deleteFlg) {  
    this.deleteFlg = deleteFlg;  
}  
public Map<String, Object> getSession() {  
    return session;  
}  
@Override  
public void setSession(Map<String, Object> session) {  
    this.session = session;  
}  
}
```

myPage.jspの作成

4 myPage.jsp



① 「プロジェクト」「WebContent」を選択し、「右クリック」「新規」「JSPファイル」を選択します。



② 「名前(M):」欄に「myPage.jsp」を入力し、完了ボタンをクリックします。

myPage.jspの作成

③ 以下の内容を写経します。

myPage.jsp(jspファイル)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="s" uri="/struts-tags"%>
<!DOCTYPE html>
<html>
<head>

<meta charset="UTF-8">
<link rel="stylesheet" type="text/css" href="./css/style.css">

<title>MyPage画面</title>

</head>
```

次へ続きます。

前の続きです。

```
<body>
  <div id="header">
  </div>
  <div id="main">
    <div id="top">
      <p>MyPage</p>
    </div>
    <div>
      <s:if test="session.message == " ">
        <h3>ご購入情報は以下になります。</h3>
        <table>
          <tr>
            <td>商品名</td>
            <td><s:property value="session.buyItem_name" /></td>
          </tr>
        </table>
      </s:if>
    </div>
  </div>
</body>
```

半角のシングルクォーテーション2個

次へ続きます。

前の続きです。

```
<tr>
    <td>値段</td>
    <td>
        <s:property value="session.total_price" />
        <span>円</span>
    </td>
</tr>
<tr>
    <td>購入回数</td>
    <td>
        <s:property value="session.total_count" />
        <span>個</span>
    </td>
</tr>
<tr>
    <td>支払い方法</td>
    <td><s:property value="session.total_payment" /></td>
</tr>
</table>
```

次へ続きます。

前の続きです。

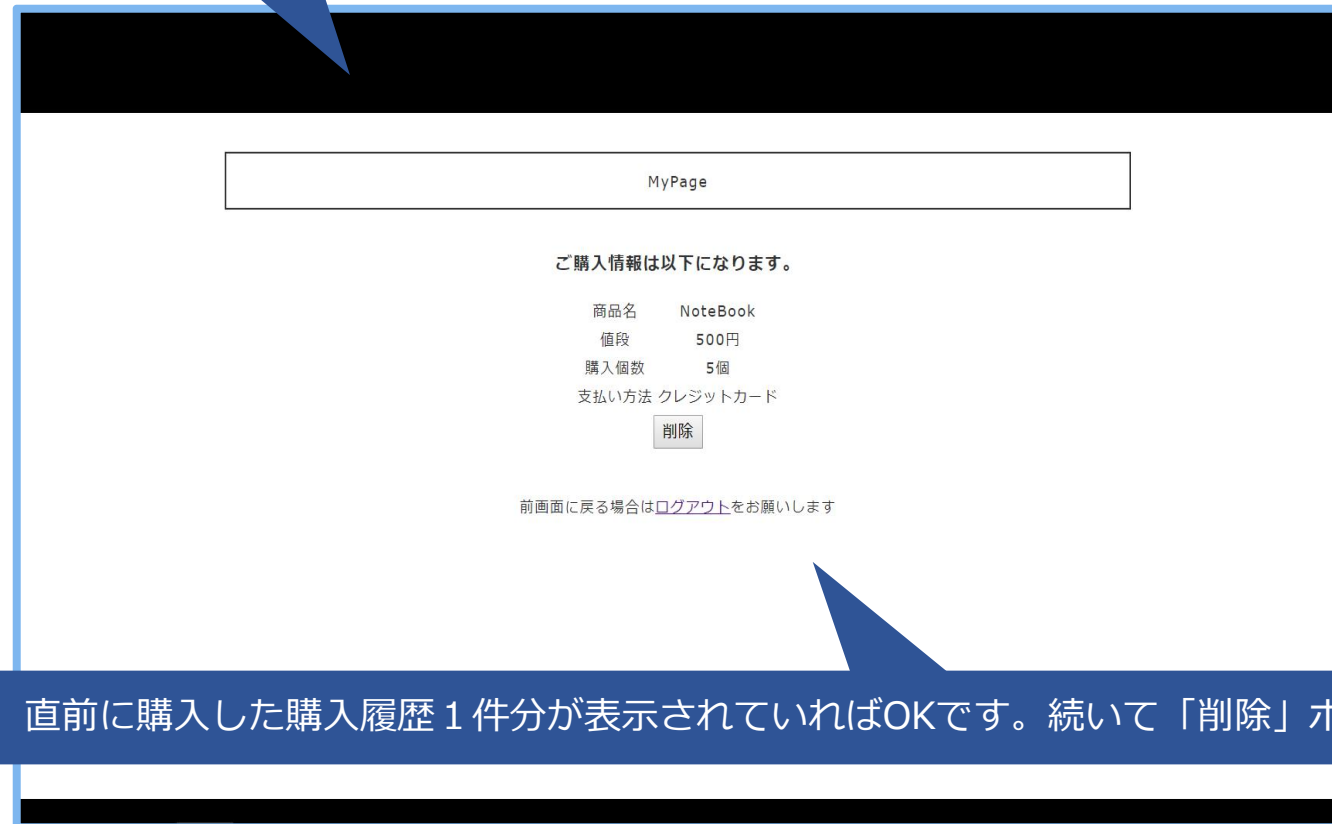
```
<s:form action="MyPageAction">
    <input type="hidden" name="deleteFlg" value="1">
    <s:submit value="削除" />
</s:form>
</s:if>
<s:if test="session.message != null">
    <h3><s:property value="session.message"/></h3>
</s:if>
<div>
    <br>
    <span>前画面に戻る場合は</span>
    <a href='<s:url action="HomeAction" />'>ログアウト</a>
    <span>をお願いします</span>
</div>
</div>
</div>
<div id="footer">
</div>
</body>
</html>
```

次へ続きます。

myPage機能の確認

myPage.jspの画面

templateプロジェクトを実行し、buyItemComplete画面の「マイページ」リンクから下記のmyPage.jsp画面が表示できるか確認します。



The screenshot shows a web page titled "MyPage" with a black header and footer. The main content area displays purchase information for a notebook. The text "ご購入情報は以下になります。" is centered above a table. The table has two columns: the first column lists the item name, unit price, quantity, and payment method, while the second column contains the corresponding values. Below the table is a "削除" (Delete) button. At the bottom, a link labeled "ログアウト" (Logout) is provided for returning to the previous page.

MyPage	
ご購入情報は以下になります。	
商品名	NoteBook
値段	500円
購入個数	5個
支払い方法	クレジットカード
<input type="button" value="削除"/>	
前画面に戻る場合は ログアウト をお願いします	

直前に購入した購入履歴 1 件分が表示されていればOKです。続いて「削除」ボタンをクリックします。

myPage機能の確認

myPage.jspの画面

こちらの画面に切り替われば成功です。

MyPage

商品情報を正しく削除しました。

前画面に戻る場合は[ログアウト](#)をお願いします

「ログアウト」のリンクからlogin.jsp画面に切り替わるかも確認します。

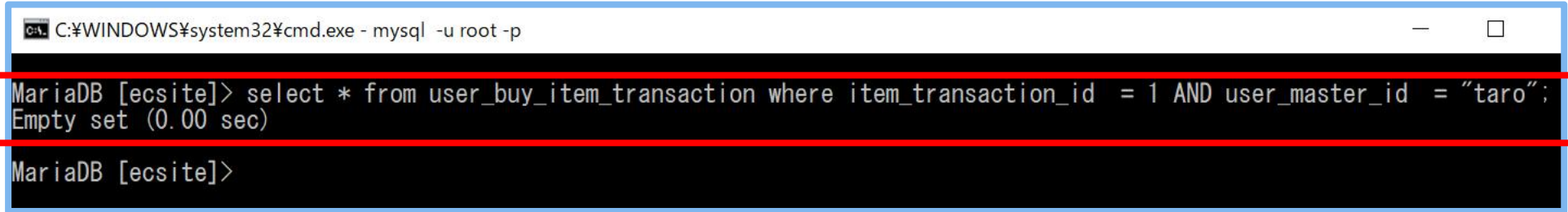
Login

商品を購入するにはログインをお願いします。

新規ユーザー登録は[こちら](#)

user_buy_item_transactionテーブル

コマンドプロンプト等を使用し、購入履歴がuser_buy_item_transactionテーブルから削除されているか確認します。

A screenshot of a Windows command prompt window. The title bar shows the path 'C:\WINDOWS\system32\cmd.exe - mysql -u root -p'. The command prompt shows a MySQL session with the prompt 'MariaDB [ecsite]>'. A red rectangular box highlights the command 'select * from user_buy_item_transaction where item_transaction_id = 1 AND user_master_id = "taro";' and its output 'Empty set (0.00 sec)'. Below the highlighted text, the prompt 'MariaDB [ecsite]>' is visible again.

```
C:\WINDOWS\system32\cmd.exe - mysql -u root -p
MariaDB [ecsite]> select * from user_buy_item_transaction where item_transaction_id = 1 AND user_master_id = "taro";
Empty set (0.00 sec)
MariaDB [ecsite]>
```

「ログインしていたユーザー」が「直前に購入した商品」の購入履歴（過去の同じ商品の購入履歴を含みます）がすべて削除されていれば成功です。