Introduction

This Project is to create a web-based educational marking system, which would generate an education record for a school. Teachers can access this database system from any kinds of internet browser. This system will provide clearly introduction, so all user can learn fast even they do not have any computer background. The major purpose of this system is to collect educational data from all courses. Principal can view all students and generate transcript for a student.

Web Design

Since this is a web-based system, its first page must be a login form. Basically, every teacher will get a teacher id and a password from the administrator of the system. When a teacher is trying to login, the system will connect to the school database. The system can verify the ID and password, provide the access to the authorized user or show the error message.

Once teachers login successfully, the system will generate the second page, which will show all the courses of the teacher this semester. Teachers can choose a specific class to manage the marks of students, and the system only
show the students in that class. The system will also provide a principal ID. When the principal login to the system, he/she will see all the students in this school. The principal can output the transcript of every student in this school.

### Database Analysis

In this project, I will use relational database with schema \{teacherName, password, teacherID, courseName, studentName, mark\}. On the login page, we have the functional dependency (FD): \text{teacherID} \rightarrow \text{password} \text{teacherName} since teacher name and password of two teachers can be same or change in some case. However, I do not think the teacher ID can be same, because teacher ID should be create by administrator when the teacher works at the first day associated with his/her photo ID and SIN number.

After a teacher logs in successfully, the system will show the courses he/she teaches this semester. This shows us another FD: \text{courseName} \rightarrow \text{teacherID} since every class has to be taught by one teacher but a teacher can teach two or more classes in one semester. If the teacher click the button of a class, the page will show him/her all students learning in this class so that the teacher can update the students’ marks. This shows the third FD: \text{courseName}studentName \rightarrow \text{mark} since the relationship between students and course is many to many.

Therefore, we get a database schema \{teacherName, password, teacherID, courseName, studentName, mark\} with a set of FD \{teacherID \rightarrow password teacherName, courseName \rightarrow teacherID, courseName studentName \rightarrow mark\} in the First Normal Form. Here we have the candidate key \{courseName,
studentName}. However, the nonprime attributes teacherID, teacherName, courseName is not fully dependent on candidate key \{courseName, studentName\}, so we decompose the original schema into two schemas \{teacherName, password, teacherID, courseName\} and \{courseName, studentName, mark\}. Therefore, their both are in the **Second Normal Form**. However, FD: teacherID→passwordteacherName violates the Third Normal Form since the nonprime attributes password and teacherName are transitively dependent upon the candidate key courseName. Therefore, we decompose it into two schemas. Finally, we get schemas \{teacherName, password, teacherID\}, \{teacherID, courseName\}, \{courseName, studentName, mark\} in the **Third Normal Form**. We can also show this relation in an acyclic hypergraph.

For the schema \{teacherName, password, teacherID\}, we can build an entity named "teacher" with primary key \{teacherID\}. For the schema \{courseName, studentName, mark\}, we can build another entity named "course" with primary key \{courseName, studentName\}. For the schema \{teacherID, courseName\}, we can build a relationship between teacher and course named "teach" with primary key \{courseName\}. Then we have the following entity relationship diagram.

According to the above diagram, we can create the relational tables in our system using MySQL syntax.

```sql
CREATE TABLE Teacher
(
    teacherID varchar(255) NOT NULL,
    teacherName varchar(255) NOT NULL,
    password varchar(255) NOT NULL,
    PRIMARY KEY (teacherID)
);
```

3
CREATE TABLE Teach
(
    teacherID varchar(255) NOT NULL,
    courseName varchar(255) NOT NULL,
    PRIMARY KEY (courseName),
    FOREIGN KEY (teacherID) REFERENCES Teacher(teacherID)
    ON UPDATE CASCADE ON DELETE RESTRICT
);

CREATE TABLE Course
(
    courseName varchar(255) NOT NULL,
    studentName varchar(255) NOT NULL,
    mark int,
    PRIMARY KEY (courseName, studentName),
    FOREIGN KEY (courseName) REFERENCES Course(courseName)
    ON UPDATE CASCADE ON DELETE RESTRICT
);

At this point, our database is ready, the next step will be using the knowledge of PHP5 and java script to make the web page dynamic.

Implementation

For my web page style, I use the technology of CSS. I link the following two file.

```html
<link rel="stylesheet" type="text/css" href = "project.css"/>
<link rel="stylesheet" type="text/css" href = "login.css"/>
```

project.css provides the background style to all pages. login.css set only the style of login page. Here I use the idea of html5 as follow.

```html
<article class="login">
<header>
    ...
</header>
<h1>LOGIN</h1>
<form action="login.php" method="post">
    ...
</form>
<footer>
    ...
</footer>
```
All pages start from an article tag. I give a class name to this article tag. Inside of the article tag, we have the header tag and footer tag. We also have a form on the login page. All styles on this page are under this "login" class as follow.

```css
article.login {
    color: #2F19C0;
    background: #80E9E9;
    width: 99.7%;
    border: 1px solid #FF06C0;
}
article.login header {
    ...
}
article.login footer {
    ...
}
```

Since this is a login page, I need to manage my login form. I also need to verify all the input format is legal. Here I use DOM level 3 to handle all event on this page. I register the event at the end of the page.

```html
<script type="text/javascript" src="loginr.js"></script>
<body>
Here I register all event by setting some IDs in tags and use the build-in function "getElementById()" as follow.

```javascript
var dom1 = document.getElementById("teacher_id");
var dom2 = document.getElementById("pwd_id");
dom1.addEventListener("blur", chkID, false);
dom2.addEventListener("blur", chkPassword, false);
``` 

At the begin of the page, I link a java script file

```html
<script type="text/javascript" src="login.js"></script>
<body>
because all functions should be defined before the event function call. Both function chkID and chkPassword will verify all input and give out the error message if not correct.

```javascript
function chkID(event){
    ...
}
function chkPassword(event){
```
Here I use the regular expression to check the input format. Since the login form redirects to the same page, this action may course the information redundancy. I use a hidden input type as a key to solve this problem.

```php
<input type="hidden" name="submitted" value="1">
```

when my hidden variable "submitted" get value '1', the program will run the php and connect the database.

```php
<?php
if($_POST["submitted"]){
    ...$con=mysqli_connect("localhost", "liang121", "123456", "liang121");
    $q = "SELECT teacherID, teacherName FROM Teacher WHERE teacherID = '$e' AND password = '$p'";
    $result = mysqli_query($con, $q);
    if(mysqli_num_rows($result)>0){
        session_start();
        $SESSION["logged_in"] = 1;
        ...header("location: marking.php");
    }
    mysqli_free_result($result);
    mysqli_close($con);
}?>
```

Here the php page run a query in MySQL, and store the input ID and user’s name into the session variable. The page will redirect to "marking.php" with a session variable.

On the marking page, my system would check the login status by

```php
<?php
    session_start();
    if($_SESSION["logged_in"] != 1){
        header("location: login.php");
    }
?>
```

if the login status is not correct, the page will go back to the login page.

when this page start, we connect the database by

```php
$q = "SELECT courseName FROM Teach
```
This query shows all the courses the teacher teaches this semester. Since we do not know how many courses the teacher will teach, we need to name IDs with variable.

```php
<?php
    if (mysqli_num_rows($result)>0){
        $i = 0;
        while($row = mysqli_fetch_assoc($result)){
            $i ++;
            <li>
                <button id="course" type="button" value=""<?php echo $i?>">"<?php echo $row["courseName"]?>"</button>
            </li>
        }
    }
?>
```

Here we use variable $i to define button’s id and register event by using for loop

```javascript
var rows = document.getElementsByTagName("li");
for(var j=1; j<=rows.length; j++){
    document.getElementById("course"+j).addEventListener("click", showTable, false);
}
```

Here we generate a dynamic page using AJAX. When the teacher clicks the course button, the system will generate a marking table on the right hand side.

In order to generate such a table, we wrote the following code in javascript
function showTable(event){
    var course = event.currentTarget.value;
    xhr = new XMLHttpRequest();
    xhr.onreadystatechange = function(){
        if(xhr.readyState==4 && xhr.status==200){
            createTable(xhr.responseText, course);
        }
    }
    xhr.open("GET", "markRecord.php?course="+course, true);
    xhr.send(null);
}

and markRecord.php in background. This php page will get the courseName from java script and connect the database

<?php
    $course=$_GET["course"]; if (strlen($course)) {
        $con=mysqli_connect("localhost", "liang12", "123456", "liang12");
        $q = "SELECT * FROM Course WHERE courseName='$course'";
        $result = mysqli_query($con, $q);
        if (mysqli_num_rows($result)!=0) {
            $json=array("studentName"=>array(), "mark"=>array());
            $i=1;
            while($row = mysqli_fetch_assoc($result)) {
                $json["studentName"][]=$row["studentName"];
                $json["mark"][]=$row["mark"];
                $i++;
            }
        }
    }
    mysqli_free_result($result);
    mysqli_close($con);
    echo json_encode($json);
}?>

Here we use a json object. Json object is a two or more dimension array. In my case, it is a two dimension array. It will store all studentNames associated with marks. It is very powerful object. Then we will receive this object to another java script function createTable()

function createTable(response, course) {
    var myArr = JSON.parse(response);
    var dom = document.getElementById("showMark");
    for (i=0; i < myArr.studentName.length; i++){
        ...
}
newTable.appendChild(newTr);
newTr.appendChild(newTd1);
newTr.appendChild(newTd2);
newTd1.appendChild(textNode1);
newTd2.appendChild(input);
inpuy.addEventListener("change",
    function(event){updateTable(event, course)},false);
}
dom.appendChild(newTable);

In order to add event listener in our dynamic object, we build a appendChild tree.

<section id="showMark">
    <table>
        <tr>
            <td>
                <input>
            </td>
        </tr>
    </table>
</section>

Then we add event listener on the input element. If the teacher changes any mark in the table, the function updateTable() will be called. It will pass the mark to the file updateMark.php. It will run the query

UPDATE Course SET mark=$mark
    WHERE courseName='$course' AND studentName='$name'

If we login in using the principal ID, we will get a select table which will show all student by running a query "SELECT DISTINCT studentName FROM Course". If user select a student and click "transcript" button, the event function of java script will pass the studentName as a variable to createTran.php file. The php file runs a query "SELECT * FROM Course WHERE studentName=$name". Then the system will create a transcript on the right hand side.

I attempted to use fpdf.php from http://www.fpdf.org/ to generate after this step, but I only could generate blank pdf page. The file system of hercules.cs.uregina.ca does not allow me to generate any pdf file.
Testing

In this project, I use the server of University of Regina, hercules.cs.uregina.ca, so my website is www2.cs.uregina.ca/ liang12l. Using teacher ID: MS100 and password: 123456ms, anyone can access to the marking.php page. Using teacher ID: Principal and password: 123456js, anyone can access to the principal.php page.

Conclusion

In this project, I used the knowledge of HTML5, CSS, PHP5, JavaScript, AJAX, JSON and MySQL to create a dynamic web page. All queries are created by PHP and run in background. Users only need to click a button in the interface, then tables will show up without loading the whole page. Any change inside input-boxes automatically update data in the database. I had tried my best to make the interface look professional and easy to use. Hopefully, this web-base system can help the educational assessment.