深 圳 大 学 实 验 报 告

课程名称：1503100001 移动设备交互应用

实验项目名称：PHP MySQL Apache Lab

学院：Computer Science and Software Engineering

专业：Software Engineering

指导教师：Basker George

报告人 ：林宪亮 学号：2022150130 班级：国际班

实验时间：2024 年 12 月 28 日

实验报告提交时间：2024 年 12 月 29 日

教务部制

实验目的与要求：(Purpose of Experiment purpose and Requirements)

Modify the PHP code to include3 the following:

1) Proper Layout

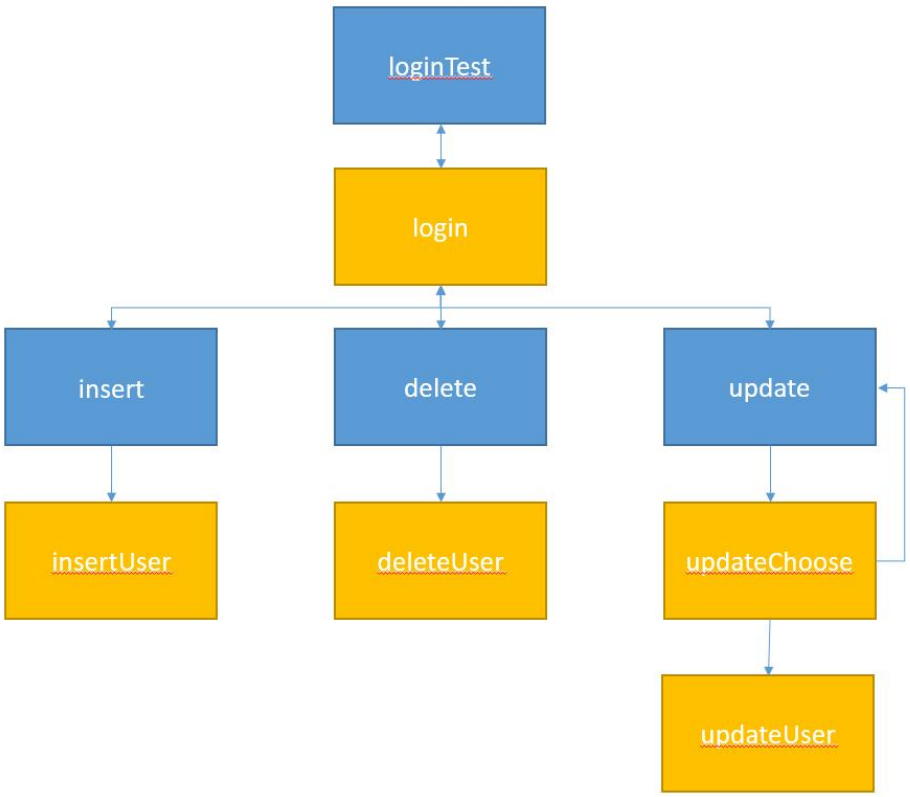
2) Header and Footer

3) CSS3 Style sheet

4) Login Page so that, only if the username and password is correct they can ADD DELETE SEARCH or UPDATE Employee Table

5) IF USER IS NOT LOGGED IN, DON'T ALLOW TO ACCESS ANY OTHER WEBPAGE ON THIS SITE

6) Create User Registration



1. 首先创建 `includes/header.php`:

```php
<?php
session_start();
// 检查用户是否登录（除了 login.php 和 register.php 页面）
$current_page = basename($_SERVER['PHP_SELF']);
if (!isset($_SESSION['logged_in']) &&
    !in_array($current_page, ['login.php', 'register.php'])) {
    header('Location: login.php');
    exit;
}
?>
<!DOCTYPE html>
<html>
<head>
    <title>员工管理系统</title>
    <link href="css/style.css" rel="stylesheet" type="text/css">
</head>
<body>
    <?php if(isset($_SESSION['logged_in'])): ?>
    <nav class="main-nav">
        <ul>
            <li><a href="dashboard.php">首页</a></li>
            <li><a href="employees.php">员工管理</a></li>
            <li><a href="logout.php">登出</a></li>
        </ul>
    </nav>
    <?php endif; ?>
    <div class="container">
```

这段代码主要是在处理用户登录验证。首先，我通过 session_start() 启动了会话功能，这样我就能在不同页面之间存储和读取数据，比如用户的登录状态。接着，我检查当前页面是不是 login.php 或 register.php，如果不是，而且用户没有登录（$_SESSION['logged_in'] 没有设置），我会把用户重定向到登录页面。
在页面加载部分，如果用户已经登录，我会显示导航菜单，包含指向首页、员工管理和登出的链接。如果用户未登录，导航菜单就不会显示。

2. 创建 `includes/footer.php`:

```php:includes/footer.php
    </div>
    <footer class="main-footer">
        <p>&copy; <?php echo date('Y'); ?> 员工管理系统. 保留所有权利.</p>
    </footer>
```

```
</body>
</html>
```

这段代码是用来显示网页底部的版权信息。首先，我用 关闭了之前的容器，然后定义了一个 <footer> 元素，用来放置页面底部的内容。里面有一个 <p> 标签，显示了版权符号和版权声明。
其中，© <?php echo date('Y'); ?> 会动态输出当前年份，这样我就不用每年手动更新年份了。其他部分则是网站的名称和版权声明，表明 "员工管理系统" 拥有该网站的所有权利。
最后，我关闭了 </body> 和 </html>，结束了页面的主体和整个文档。

3. 修改登录页面 `login.php`:

```php:login.php
<?php
require_once 'includes/header.php';
if (isset($_POST['username']) && isset($_POST['password'])) {
    $conn = new mysqli('localhost', 'root', '', 'abc_company');

    if ($conn->connect_error) {
        die("连接失败: " . $conn->connect_error);
    }

    // 使用密码哈希进行比较
    $stmt = $conn->prepare("SELECT id, password FROM users WHERE username = ?");
    $stmt->bind_param("s", $_POST['username']);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows == 1) {
        $user = $result->fetch_assoc();
        if (password_verify($_POST['password'], $user['password'])) {
            $_SESSION['logged_in'] = true;
            $_SESSION['user_id'] = $user['id'];
            $_SESSION['username'] = $_POST['username'];

            header('Location: dashboard.php');
            exit;
        }
    }
    $error_msg = '用户名或密码错误';

    $conn->close();
```

```
}
?>
<div class="login-container">
    <h1>员工信息管理系统</h1>
    <div class="login-box">
        <h2>登录</h2>
        <?php if(isset($error_msg)): ?>
            <div class="error"><?php echo $error_msg; ?></div>
        <?php endif; ?>
        <form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
            <div class="form-group">
                <label>用户名:</label>
                <input type="text" name="username" required>
            </div>
            <div class="form-group">
                <label>密码:</label>
                <input type="password" name="password" required>
            </div>
            <button type="submit">登录</button>
        </form>
        <p class="register-link">
            还没有账号? <a href="register.php">立即注册</a>
        </p>
    </div>
</div>
<?php require_once 'includes/footer.php'; ?>
```

这段代码处理了用户登录的功能。当用户提交了用户名和密码时，我首先通过 mysqli 连接到数据库。如果连接失败，我会停止执行并显示错误信息。接着，我准备一个 SQL 查询，查找与用户名对应的 id 和 password。我使用了预处理语句和参数绑定来避免 SQL 注入问题。

如果查询结果中找到了用户，我就通过 password_verify 比对输入的密码和数据库中存储的密码哈希。如果密码正确，我会将用户的登录状态存储在 $_SESSION 中，并将用户重定向到后台首页。如果密码错误，我就显示一个错误消息。最后，页面展示了一个登录表单，用户可以在其中输入信息，或者点击链接跳转到注册页面。

4. 创建注册页面 `register.php`:

```php:register.php
<?php
require_once 'includes/header.php';
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $conn = new mysqli('localhost', 'root', '', 'abc_company');
```

```php
    if ($conn->connect_error) {
        die("连接失败: " . $conn->connect_error);
    }

    // 检查用户名是否已存在
    $stmt = $conn->prepare("SELECT id FROM users WHERE username = ?");
    $stmt->bind_param("s", $_POST['username']);
    $stmt->execute();

    if ($stmt->get_result()->num_rows > 0) {
        $error_msg = '用户名已存在';
    } else {
        // 创建新用户
        $hashed_password = password_hash($_POST['password'], PASSWORD_DEFAULT);
        $stmt = $conn->prepare("INSERT INTO users (username, password) VALUES (?, ?)");
        $stmt->bind_param("ss", $_POST['username'], $hashed_password);

        if ($stmt->execute()) {
            $_SESSION['register_success'] = true;
            header('Location: login.php');
            exit;
        } else {
            $error_msg = '注册失败，请重试';
        }
    }

    $conn->close();
}
?>
<div class="login-container">
    <h1>员工信息管理系统</h1>
    <div class="login-box">
        <h2>注册新用户</h2>
        <?php if(isset($error_msg)): ?>
            <div class="error"><?php echo $error_msg; ?></div>
        <?php endif; ?>
        <form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="post">
            <div class="form-group">
                <label>用户名:</label>
                <input type="text" name="username" required>
            </div>
            <div class="form-group">
                <label>密码:</label>
```

```
                <input type="password" name="password" required>
            </div>
            <div class="form-group">
                <label>确认密码:</label>
                <input type="password" name="confirm_password" required>
            </div>
            <button type="submit">注册</button>
        </form>
        <p class="login-link">
            已有账号? <a href="login.php">立即登录</a>
        </p>
    </div>
</div>
<?php require_once 'includes/footer.php'; ?>
```

在这段 PHP 代码中，我实现了一个用户注册功能。首先，我通过 require_once 引入了页面的头部文件 header.php。接着，我检查请求方法是否为 POST，如果是，表示用户提交了注册表单。

在处理用户输入时，我首先通过 new mysqli 连接到 MySQL 数据库 abc_company。若连接失败，程序会显示连接错误信息并终止。之后，我准备了一个 SQL 查询语句，检查用户输入的用户名是否已经存在于数据库中。通过 bind_param 方法绑定用户提交的 username 参数并执行查询。如果查询结果显示已有相同的用户名，则显示错误信息 '用户名已存在'。

如果用户名可用，接下来我对用户输入的密码进行加密处理，使用 PHP 的 password_hash 函数生成哈希值。然后，我再次准备一个 SQL 插入语句，将新用户的用户名和加密后的密码插入到数据库中。成功插入数据后，设置会话变量 register_success 为 true，并通过 header 函数重定向到登录页面 login.php。如果注册过程出现任何错误，我会显示一个 '注册失败，请重试' 的错误信息。

最后，我在页面的 HTML 部分中通过 $_SERVER['PHP_SELF'] 将表单的提交路径指向当前页面。如果有错误信息，就会在页面上显示相应的提示。表单中包括用户名、密码和确认密码输入框。通过表单提交后，页面将再次进行处理。

总体来说，这段代码实现了一个基本的注册功能，包含了数据库操作、表单验证以及简单的错误处理。

5. 更新样式表 `css/style.css`:

```css:css/style.css
/* 保留原有样式并添加以下内容 */
.container {
    width: 90%;
    margin: 0 auto;
    padding: 20px;
}
.main-nav {
```

```
    background: #333;
    padding: 10px 0;
}
.main-nav ul {
    list-style: none;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
}
.main-nav li {
    margin: 0 15px;
}
.main-nav a {
    color: white;
    text-decoration: none;
}
.login-container {
    max-width: 400px;
    margin: 50px auto;
}
.error {
    color: red;
    margin: 10px 0;
    padding: 10px;
    background: #ffe6e6;
    border: 1px solid #ff9999;
}
.form-group {
    margin-bottom: 15px;
}
.main-footer {
    text-align: center;
    padding: 20px;
    background: #f5f5f5;
    margin-top: 50px;
}
```

在这段 CSS 代码中，我主要定义了一些样式来美化页面的布局和元素。
首先，我为 .container 设置了一个宽度为 90% 的页面容器，并通过 margin: 0 auto 来居中显示，padding: 20px 增加了容器的内边距，使内容与容器边缘保持一定距离。
接着，我设计了一个 .main-nav 类，它表示页面的主导航栏。我为它设置了深色的背景(background: #333) 和上下各 10px 的内边距，使其更加显眼。在 .main-nav ul 中，我通

过 list-style: none 去除了默认的列表项样式，设置了 margin 和 padding 为 0，使用 display: flex 来让列表项水平排列，justify-content: center 使它们在容器中居中对齐。
在 .main-nav li 中，我为每个列表项添加了左右 15px 的外边距，确保导航项之间有适当的间隔。而 .main-nav a 则设置了白色的字体颜色 (color: white) 并去除了链接的下划线 (text-decoration: none) 以保持简洁的外观。
对于登录区域，我为 .login-container 设置了最大宽度为 400px，并通过 margin: 50px auto 将它居中，并且在上方和下方留出了 50px 的空间。
如果有错误信息显示，.error 类会应用到错误提示框上。我设置了红色的字体颜色 (color: red)，背景色为浅红色 (background: #ffe6e6)，并用 border: 1px solid #ff9999 加上了边框，使错误信息更加醒目。
.form-group 为表单的每个输入组提供了 15px 的底部间距，确保表单项之间的间隔合理。
最后，.main-footer 类用于定义页面的底部区域。我让文本居中显示 (text-align: center)，并设置了 20px 的内边距，同时背景颜色为浅灰色 (background: #f5f5f5)，并通过 margin-top: 50px 为底部留出了空间。
总体来说，这段 CSS 代码主要实现了页面的布局、导航栏的样式、表单的美化以及错误信息的提示样式，使得页面更加整洁和用户友好。

6. 创建数据库表:

```sql
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

这段 SQL 代码的作用是创建一个名为 users 的表，用来存储用户的信息。我定义了四个字段：
id INT AUTO_INCREMENT PRIMARY KEY
我为表定义了一个 id 字段，用来唯一标识每个用户。INT 表示这个字段存储的是整数，AUTO_INCREMENT 意味着每次插入新记录时，id 会自动递增，因此不需要手动指定。PRIMARY KEY 保证了每个 id 的唯一性，使其成为表中每一行数据的主键。
username VARCHAR(50) UNIQUE NOT NULL
我为用户设置了一个 username 字段，它的类型是 VARCHAR(50)，意味着用户名最多可以包含 50 个字符。UNIQUE 确保每个用户名都是唯一的，避免了重复用户名的情况。NOT NULL 约束保证每个用户必须提供用户名，不能为空。
password VARCHAR(255) NOT NULL
我为每个用户设计了一个 password 字段，用来存储用户的密码。VARCHAR(255) 表示密码最长可以存储 255 个字符。NOT NULL 约束要求每个用户必须有密码，不能为空。
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
我还创建了一个 created_at 字段，用来记录每个用户账户的创建时间。TIMESTAMP 类型表示存储时间信息，DEFAULT CURRENT_TIMESTAMP 表示如果插入数据时没有指

定时间，系统会自动使用当前时间作为默认值。
总结来说，这段代码定义了一个用户表，用来存储用户的唯一 ID、用户名、密码以及创建时间。通过这些字段的约束，确保了数据的完整性和一致性。

界面：

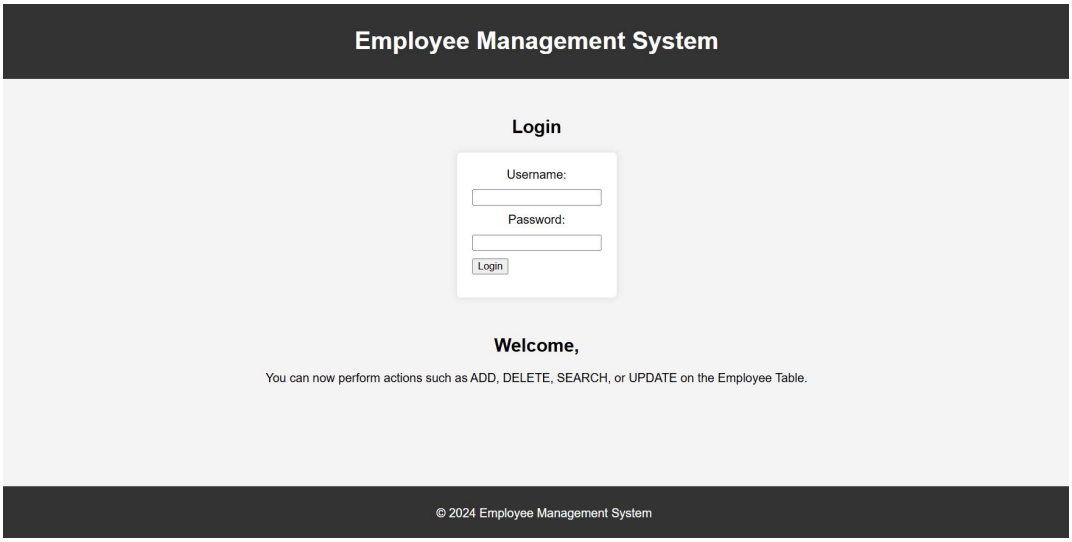


图 1 登录页

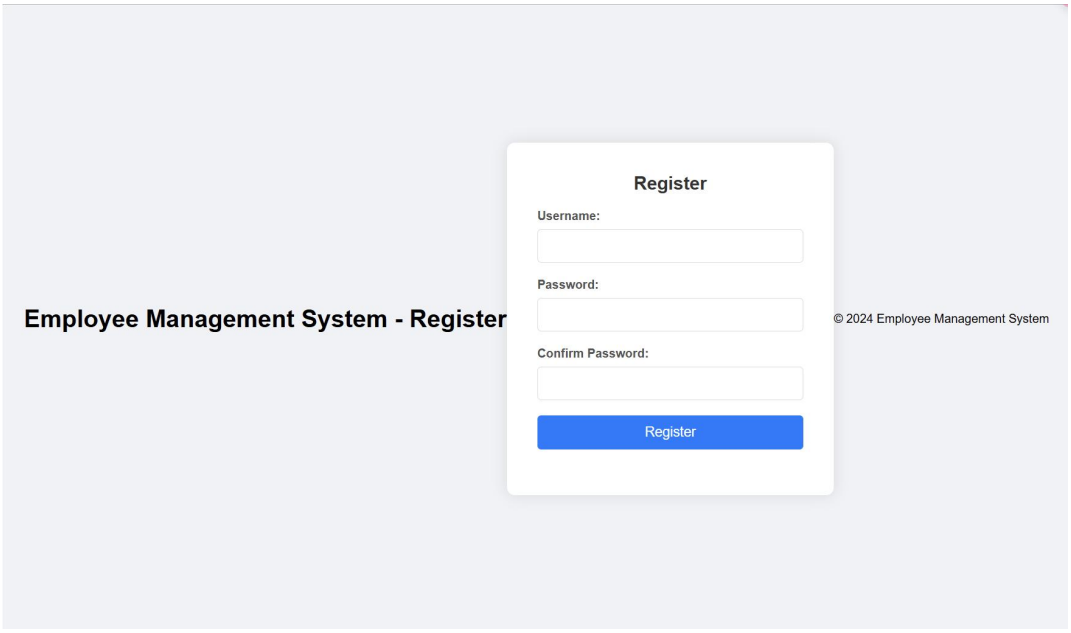


图 2 注册页

数据处理分析：(**Experiment process and content)**

1. 用户登录验证流程

- 在`includes/header.php`中，通过`session_start()`启动会话，获取当前页面名称`$current_page`，检查用户是否登录（除`login.php`和`register.php`外），未登录则重定向到`login.php`。这确保了用户在访问其他页面之前必须先登录，增强了系统的安全性和数据的保密性。

- 在`login.php`中，当用户提交用户名和密码时，使用`mysqli`连接数据库。通过预处理语句`$stmt = $conn->prepare("SELECT id, password FROM users WHERE username =?")`查询用户信息，避免了 SQL 注入问题。若查询到用户且密码通过`password_verify`验证正确，将用户登录状态及相关信息存储在`$_SESSION`中并跳转到`dashboard.php`，否则显示错误消息。这一流程有效地管理了用户的登录过程，保证了只有合法用户能够进入系统操作员工数据。

2. 用户注册功能实现

- 在`register.php`中，同样引入`header.php`进行登录验证。当用户提交注册表单（`REQUEST_METHOD`为`POST`）时，连接数据库并检查用户名是否已存在。若不存在，使用`password_hash`对密码进行加密处理后，将新用户信息插入数据库。注册成功后设置`register_success`会话变量并跳转到`login.php`，失败则显示相应错误消息。这一过程保证了用户注册的安全性和数据的完整性，每个用户都有唯一的用户名，密码以加密形式存储，提高了系统的安全性。

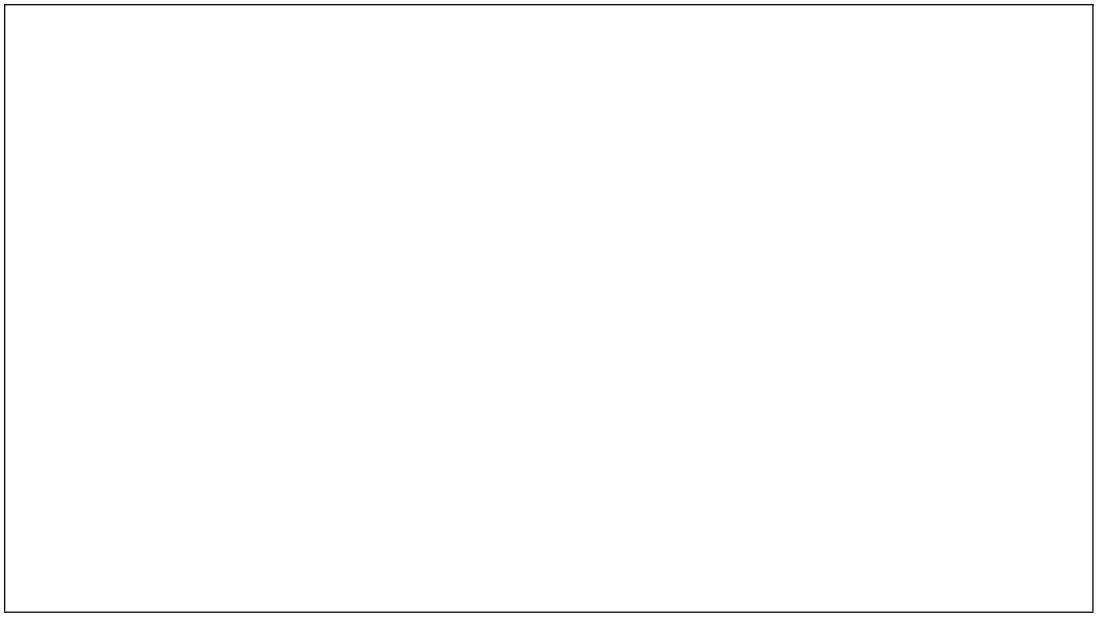
3. 页面布局与样式设计

- `css/style.css`文件定义了一系列样式规则。`.container`类设置了页面容器的宽度、居中显示及内边距，使内容布局整齐。`.main-nav`及其相关样式定义了主导航栏的外观，包括背景色、内边距、列表样式、对齐方式等，提升了导航的可视性和易用性。`.login-container`控制登录区域的最大宽度和居中显示，优化了登录界面的布局。`.error`类用于突出显示错误信息，增强了用户对错误的感知。`.form-group`为表单输入组提供合理间距，方便用户操作。`.main-footer`定义了页面底部版权信息区域的样式，包括文本对齐、内边距和背景色等，完善了页面的整体外观。

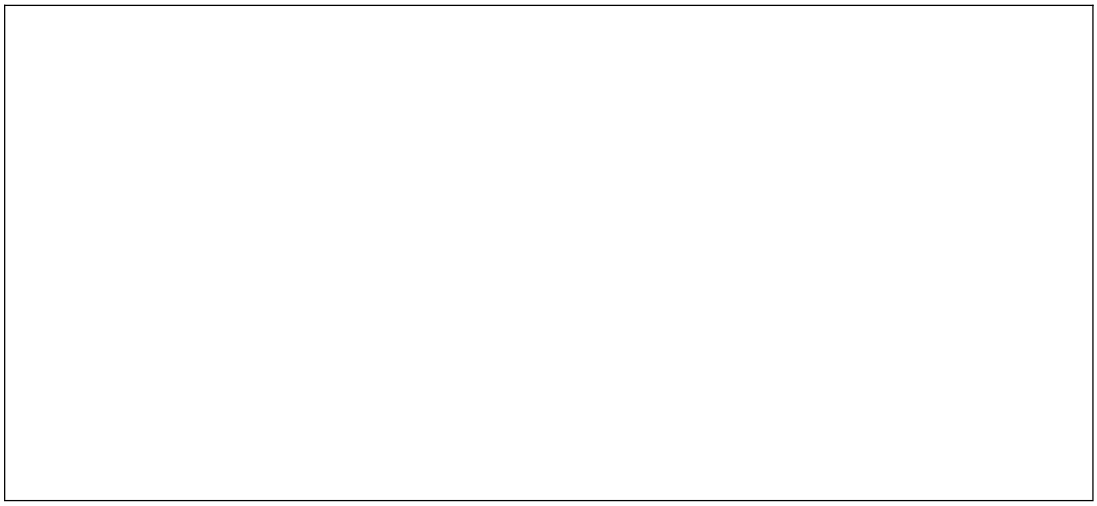
4. 数据库表结构设计

- 通过`CREATE TABLE users`语句创建了名为`users`的数据库表，包含`id`（用户唯一标识，自动递增的整数主键）、`username`（唯一且不为空的字符串，最长 50 字符）、`password`（不为空的字符串，最长 255 字符）和`created_at`（记录用户账户创建时间的时间戳，默认当前时间）四个字段。这种结构确保了用户数据的完整性和一致性，每个用户具有唯一的标识，用户名不可重复，密码有足够的存储长度，创建时间自动记录，方便数据管理和查询。

总体而言，本实验通过上述代码实现了一个具有用户登录验证、注册功能、合理页面布局与样式以及完善数据库表结构的员工管理系统基础架构。各个部分相互协作，有效地管理用户访问权限，保证数据的安全性和完整性，同时提供了较好的用户界面体验。然而，系统可能还需要进一步优化和扩展，例如添加更多员工管理功能、完善错误处理机制、增强页面交互性等，以满足实际业务需求。 在实际应用中，可根据具体业务场景对系统进行定制和完善。例如，在员工管理功能方面，可进一步细化员工信息的录入、查询、修改和删除操作，增加数据验证和提示功能，确保员工数据的准确性和完整性。



深圳大学学生实验报告用纸



指导教师批阅意见： 成绩评定： 指导教师签字： 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。