有关(*)项的实现原理的文档说明

(1)防止SQL注入

在本次pj中，php防止sql注入的主要实现原理是预处理和参数化查询。

```
function insertUserData($username, $password, $phone, $email, $address, $gender, $birthday, $salt) {
    $sql = "INSERT INTO user (username, password, phone, email, address, gender, birthday, pocket, salt) VALUES
    (?, ?, ?, ?, ?, ?, ?, 0.00, ?)";
    $params = array($username, $password, $phone, $email, $address, $gender, $birthday, $salt);
    $result = dml($sql, $params);
    return ($result > 0);
}
```

如`insertUserData()`函数中的sql查询语句使用了占位符(?)来代替实际的参数值，然后将参数添加到一个数组$params中，在调用`dml()`函数时将sql查询语句和参数数组一起传递过去。

```
function dml($sql, $params = []) {
    $conn = mysqli_connect(HOST, USER, PWD, DBNAME) or die("连接或选择数据库失败！");
    // 使用预处理语句
    $stmt = mysqli_prepare($conn, $sql);
    if (count($params) > 0) {
        // 获取参数类型字符串
        $types = '';
        foreach ($params as $param) {
            if (is_int($param)) {
                $types .= 'i';
            } elseif (is_float($param)) {
                $types .= 'd';
            } else {
                $types .= 's';
            }
        }
        // 将参数绑定到查询语句中
        mysqli_stmt_bind_param($stmt, $types, ...$params);
    }
    mysqli_stmt_execute($stmt);
    $affected_rows = mysqli_affected_rows($conn);
    $id = mysqli_insert_id($conn);
    mysqli_stmt_close($stmt);
    mysqli_close($conn);
    return !empty($id) ? $id : $affected_rows;
}
```

在`dml()`函数中，使用`mysqli_prepare()`函数创建了一个mysqli_stmt对象，并将sql查询语句与参数分离。然后，使用`mysqli_stmt_bind_param()`函数将参数绑定到查询语句中。在这个过程中，参数值会被转义并包含在查询语句中，从而避免了 SQL 注入攻击。最后，执行查询语句使用`mysqli_stmt_execute()`函数进行查询，而不是直接执行 SQL 查询语句，也能够帮助防止 SQL 注入攻击。

(2)、部署。

我的pj部署在华为云服务器上，部署服务器公网IP为`123.249.47.195`。

我部署的主要实现步骤为华为云官网申请云服务器(我申请的云服务器为官网试用，试用期为一个月)，获取云服务器IP，使用vscode连接到云服务器，这里需要在vscode上安装部分插件，接下来的步骤为：

1.安装nginx：

在云服务器上安装nginx需要以下步骤：

1. 更新包管理器：sudo apt-get update
2. 安装nginx：sudo apt-get install nginx
3. 检查nginx是否成功安装：nginx -v
4. 启动nginx服务：sudo service nginx start
5. 检查nginx服务状态：sudo service nginx status
6. 在浏览器中输入服务器的IP地址或者域名，如果看到一个“Welcome to nginx!”的页面，那么说明nginx已经成功安装并运行。

注意：以上命令是在Ubuntu操作系统下执行的，如果您使用的是其他操作系统，需要相应地修改命令。

2.安装php及相关拓展：

1. 使用以下命令更新软件包列表：

```
Copy Code
sudo apt update
```

2. 然后使用以下命令安装PHP-FPM：

```
Copy Code
sudo apt install php8.0-fpm
```

如果您的系统中没有PHP 8.0，则可以使用适用于您系统的其他版本号替换 `8.0`。

3. 安装完成后，您可以使用以下命令启动PHP-FPM服务：

```
Copy Code
sudo systemctl start php8.0-fpm
```

4. 最后，您需要检查Nginx配置文件以确保其正确指向PHP-FPM的socket文件路径。通常情况下，此路径为 `/run/php/php8.0-fpm.sock`。如果此路径不同，请相应地更新Nginx配置并重新加载Nginx。

3.安装mysql：

安装MySQL通常需要以下步骤：

1. 添加MySQL的APT（Advanced Packaging Tool）软件包源

```
sudo apt-get update
sudo apt-get install mysql-server
```

2. 安装MySQL服务器软件包

```
sudo apt-get install mysql-server
```

3. 启动MySQL服务

```
sudo service mysql start
```

4. 设置MySQL管理员密码

```
sudo mysql_secure_installation
```

5. 使用MySQL命令行实用程序进行测试

```
sudo mysql
```

以上是在Ubuntu或Debian Linux上安装MySQL的一般步骤。 如果您使用其他Linux发行版或操作系统，则可能需要采取不同的措施。

注意：在启动MySQL之前，确保您的服务器已经更新并且所有必要的防火墙端口已打开以允许MySQL的流量通过。

4.将pj代码上传到服务器上，这里建议将html/css/js/php代码上传到var/www/html路径下。

5.修改相关配置文件nginx.conf，如：

```
server {
    listen 80;
    server_name 123.249.47.195;
    root /var/www/html;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }

    location /css/ {
        add_header Cache-Control "public,max-age=86400,immutable";
        expires 1d;
        access_log off;
    }

    location ~ \.php$ {
        #root /home/pj/front;
        fastcgi_pass unix:/var/run/php/php8.1-fpm.sock; # 请根据您的 PHP 版本进行相应的更改
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        include fastcgi_params;
    }
}
```

大致的实现步骤就是这样，但具体实现过程中大概率会出bug，我在部署过程中就因为php未配置mysqli扩展而一直连接不上mysql数据库，这里建议通过查看nginx的日志文件(/var/log/nginx/error.log)来排除bug的产生。

(3)、用户画像。

我的做法是结合艺术品的访问量和用户的访问记录为每件艺术品进行打分，即`艺术品评分=访问量打分+访问记录打分`，将评分最高的10件艺术品展示在“猜你喜欢”区域。具体的评分规则为首先获取全部艺术品的数据并获取全部艺术品中最高的访问量，遍历全部艺术品数据，为其加分——(当前艺术品访问量)*5/(最高的艺术品访问量)，这一打分设计的主要考虑是如果用户为新注册用户，其未产生访问记录，因此希望通过这个基础分为新注册的用户也可以推荐艺术品；其次获取当前登录用户针对每个艺术品的访问记录，如果有访问记录则为艺术品进行相应加分，这里考虑到不同访问记录的访问时间不同，相应的加分也因有所不同，此处具体实现规则，访问时间与当前时间间隔小于1天，加5分；大于1天小于3天，加4分；大于3天小于7天，加3分；大于7天小于一个月，加2分；大于一个月，加一分。具体实现参考代码。

(4)、点击过快。

```
//登录验证
const submitBtn=document.querySelector('#submit')
const thresholdTime=3000;
var disabled=true;
let timerId;
submitBtn.addEventListener('click',event=>{
    // 阻止表单默认提交行为
    event.preventDefault();
    if(!disabled){
        alert("请勿频繁操作！");
        submitBtn.disabled=true;
        return;
    }
    clearTimeout(timerId);
    disabled=false;
    timerId=setTimeout(()=>{
        disabled=true;
        submitBtn.disabled=false;
    },thresholdTime);
```

以登录为例，用户点击登录按钮后,将设置disabled=false,同时开启一个计时器，这里设置的时间阈值为3秒，用户再次点击后由于disabled=false，因此将弹出提示信息“请勿频繁操作”，同时设置submitBtn.disabled=true，使得用户不能再次点击，即拒绝用户操作。只有当计时器计时结束后，submitBtn.disabled才会再次为true，用户才可以再次点击登录按钮。