Защита от XSS

Проверять все данные, поступившие извне, на соответствие формату и санитизовать эти данные непосредственно перед выдачей клиентам.

Пример по сантизации данных с помощью фильтрации (preg_match, strip_tags):

```
function v_name($name){
  if(preg_match("/[^(\w)|(\x7F-\xFF)|(\s)]/",trim($name)) || (strlen(trim($name))<2 || strlen(trim($name))>50)){
    return false;
  }
  return true;
}
```

```
strip_tags($_COOKIE['login']),
strip_tags($_COOKIE['pass']));
```

Пример экранирования данных с помощью htmlspecialchars:

```
$name=htmlspecialchars($data[$i]['name']);
$email=htmlspecialchars($data[$i]['email']);
```

Защита от SQL Injection

Использование подготовленных запросов (PDO):

```php
function db() {
    $user='u52876';
    $pass='9106944';
    return new PDO ("mysql:host=localhost;dbname=u52876", $user, $pass,
}
```

Проверять данные на соответствие формату и экранировать кавычки и спецсимволы в поступающих от пользователя данных перед использованием их в SQL-запросе (mysql_real_escape_string):

```php
$query = sprintf("SELECT * FROM users WHERE user='%s' AND password='%s'",
            mysql_real_escape_string($user),
            mysql_real_escape_string($password));
```

Защита от CSRF

Генерация CSRF-токена:

```php
$_SESSION['token'] = bin2hex(random_bytes(24));
```

Проверка токена:

```php
if (hash_equals($_SESSION['token'], $_POST['token'])) {
```

Защита от Upload и Include

Для обеспечения защиты от Upload и Include атак можно использовать полный путь к файлу, вместо относительного:

```php
//Включение файла с использованием полного пути
include(dirname(__FILE__). '/bd.php');
include(dirname(__FILE__). '/sort.php');
```