1. XSS (межсайтовый скриптинг) - защита от вставки вредоносного кода в веб-страницы, которые могут быть использованы для кражи данных пользователя. Для защиты от XSS рекомендуется использовать фильтрацию входных данных и экранирование вывода. Пример кода:
// Фильтрация входных данных

```
function filter_input_data($input) {
    return htmlspecialchars(trim($input), ENT_QUOTES, 'UTF-8');
}
```

//Экранирование вывода

```
function filter_output_data($output){
    return htmlentities($output, ENT_QUOTES, 'UTF-8');
}
```

2. SQL Injection - защита от вставки вредоносного SQL-кода в запросы к базе данных, который может привести к краже данных пользователя. Для защиты от SQL Injection рекомендуется использовать подготовленные запросы и фильтрацию входных данных. Пример кода:

```
$stmt = $db->prepare("INSERT INTO application SET name=?,email=?,year=?,gender=?,limbs=?,biography=?");
$stmt -> execute(array(filter_input_data($_POST['fio']),filter_input_data($_POST['email']),filter_input_
```

3. CSRF (межсайтовая подделка запроса) - защита от подделки запросов, которые отправляются с других сайтов и могут привести к изменению данных пользователя. Для защиты от CSRF рекомендуется использовать токены CSRF и проверку referer. Пример кода:

```
session_start();
if (!isset($_SESSION['csrf_token'])) {
    $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
}
$token = $_SESSION['csrf_token'];
```

```
  <form action="index.php"
    method="POST">
<input type="hidden" name="token" value="<?= $token; ?>">
    <label>
```

```
if ($_SESSION['csrf_token'] !== $_POST['token']) {
  die('Invalid CSRF token');
}
if (parse_url($_SERVER['HTTP_REFERER'], PHP_URL_HOST) !== 'u53001.kubsu-dev.ru') {
  die('Invalid referer');
}
```

4. Include - защита от включения вредоносного кода из внешних файлов, которые могут привести к краже данных пользователя. Для защиты от Include рекомендуется использовать только относительные пути и проверку наличия файла. Пример кода:

```
  if (file_exists('form.php')) {
  include('form.php');
}
```

5. Upload - защита от загрузки вредоносных файлов на сервер, которые могут привести к краже данных пользователя. Для защиты от Upload рекомендуется проверять тип и размер загружаемого файла, а также использовать

уникальные имена файлов. В задании не требуется заливать файлы на сервер, поэтому приведу абстрактный пример. Пример кода:

```
// Проверка типа и размера файла
if ($_FILES['file']['type'] !== 'image/jpeg' || $_FILES['file']['size'] > 1000000)
{ die('Invalid file type or size'); }
// Генерация уникального имени файла
$filename = uniqid() . '.jpg';
move_uploaded_file($_FILES['file']['tmp_name'], 'uploads/' . $filename);
```