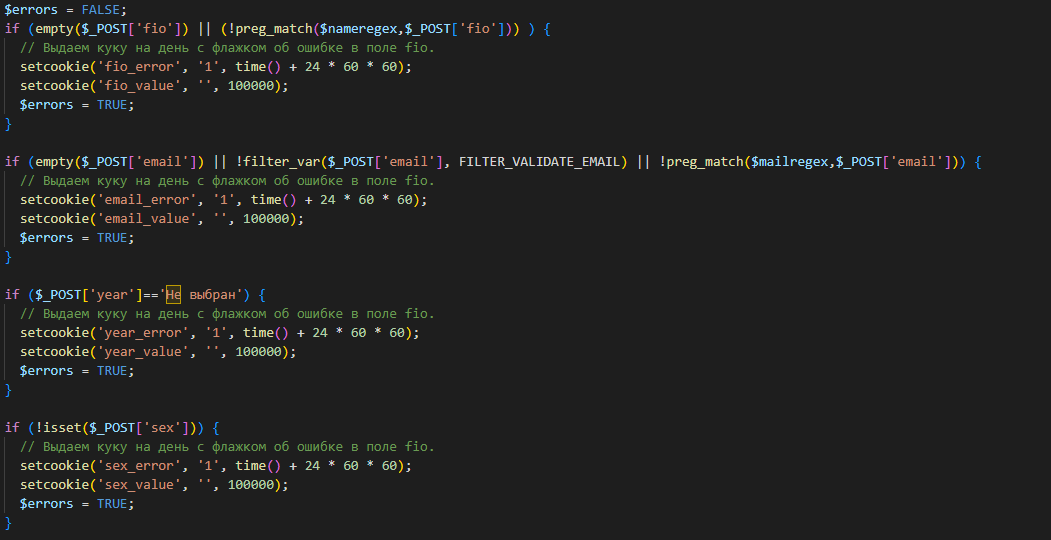
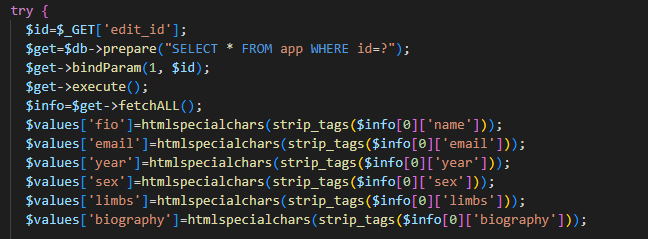
**Аудит безопасности**

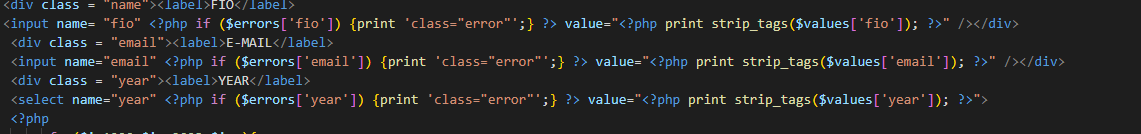
* 1. **Уязвимости XSS:**

**– Уязвимость передачи JS скрипта**

Чтобы предотвратить передачу данных, которые не удовлетворяют условиям, я ввёл проверку на соответствие передаваемых данных запрошенным и завалидовал переменные, использовал strip\_tags(), htmlspecialchars(), preg\_match(), а также доп. Проверки данных:

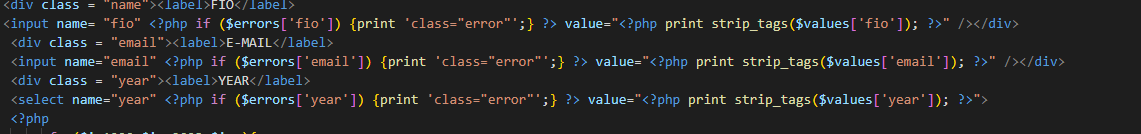


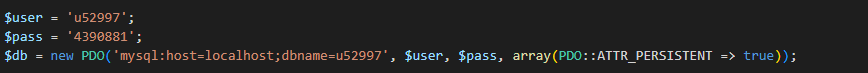


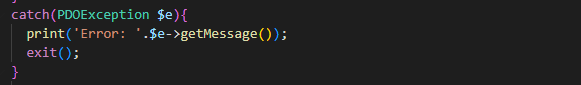


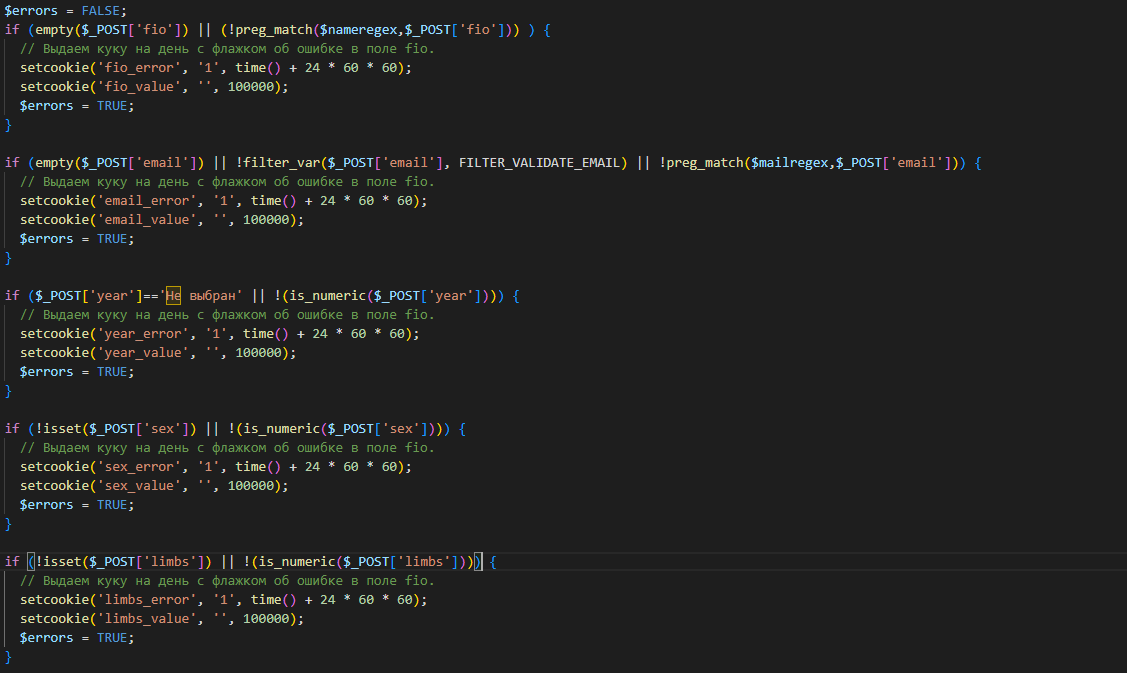
* 1. **Уязвимости SQL Injection:**

Для предотвращения уязвимостей SQL Injection я использовал подготовленные запросы (PDO) и проверял данные на соответствие с помощью функции filter\_var() и is\_numeric():



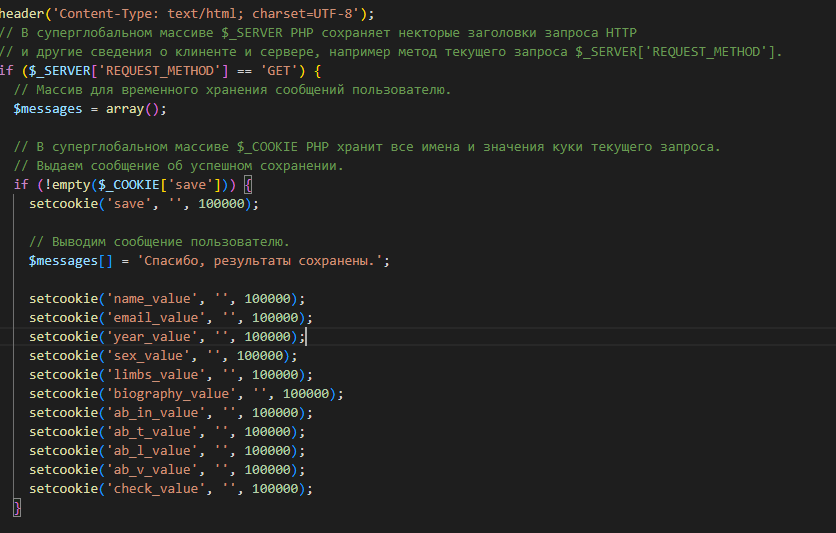






* 1. **Уязвимости CSRF**

Для защиты от уязвимости CSRF я по назначению использовал методы POST и GET, а также реализовал устаревание куки-файлов:



* 1. **Include и Upload уязвимости**

В моём коде не использовались фреймворки, общий доступ имеет только админ, которому нужно авторизоваться по логину и паролю: