<https://github.com/DimVechorko/guestBookOOPsql/blob/mysqli/classes.php>

1. Всю работу с базой данных можно организовать в одном классе.(методы: connect, put, get)
2. Также и с Валидацией. Не стоит создавать новые классы под валидацию каждого элемента. Как следствие – куча объектов в коде.Стоит записать необходимые методы в один большой класс Validation, а еще круче расширить это все до полной работы с формой.
3. Не используются свойства классов, все методы таким образом являются статическими, ну а это совсем не ООП☺.
4. Все названия классов начинаются с большой буквы. Cтоит придерживаться одного стиля($user\_name, но $strEmailAddress)
5. Я бы засунул все классы в отдельную папку.
6. Ну и сделал бы класс под капчу☺.
7. Мне нравится использовать \_\_construct внутри класса.

[https://github.com/YDziashkevich/php\_/edit/master/classGB/](https://github.com/YDziashkevich/php_/edit/master/classGB/classes/Form.php)

1. По мне, так все круто☺. Разве что можно придраться к некоторым имена переменных, плана $attrForm, но это уж совсем придирки.
2. Не помешало бы немного комментов в index.php
3. Мое личное мнение, что в index ‘e стоит писать только всевозможные подключения. Но для маленького проекта не критично.
4. Больше не к чему придраться, не хватает знаний☺.

<https://github.com/zlucan/g_book_oop/blob/master/index.php>

1. Функции прямо в index.php. Стоит над этим поработать☺.
2. Я бы перенес все файлы по тематикам. (javascript.js в js, а не в oop)
3. В целом не совсем понятно, работает ли все это.

Особенно в этой части: https://github.com/zlucan/g\_book\_oop/blob/master/inc/oop/validated.php

1. Не хватает комментариев, как все работает.
2. Семантика. Объект $o не совсем говорит о том, что это за объект.
3. Выкинуть пару «мусорных» файлов(сам этим грешу)
4. Ну и если ООП, то ООП. Никаких функций.

P.S. Без обид, с радостью приму замечания к моему коду.