**Дипломная работа по курсу**[**«PHP/SQL: back-end разработка и базы данных»**](http://netology.ru/programs/php-sql)

Разработать типовой сервис вопросов и ответов. Должна быть реализована клиентская часть сервиса и интерфейс администратора.

**Описание клиентской части**

* Пользователи могут просматривать категории, вопросы и ответы.
* Любой пользователь может задать вопрос, указав своё имя, адрес электронной почты, выбрав категорию и написав текст вопроса.
* Вопросы без ответов не публикуются на сайте.

**Вход в интерфейс администратора**

* Для попадания в интерфейс администратора нужно ввести логин и пароль.
* По умолчанию создан единственный администратор с логином admin и паролем admin.

**Возможности в интерфейсе администратора**

* Просматривать список администраторов.
* Создавать новых администраторов.
* Изменять пароли существующих администраторов.
* Удалять существующих администраторов.
* Просматривать список тем. По каждой теме в списке видно сколько всего вопросов в ней, сколько опубликовано, сколько без ответов.
* Создавать новые темы.
* Удалять существующие темы и все вопросы в них.
* Просматривать вопросы в каждой теме. По каждому вопросу видно дату создания, статус (ожидает ответа / опубликован / скрыт).
* Удалять любой вопрос из темы.
* Скрывать опубликованные вопросы.
* Публиковать скрытые вопросы.
* Редактировать автора, текст вопроса и текст ответа.
* Перемещать вопрос из одной темы в другую.
* Добавлять ответ на вопрос с публикацией на сайте, либо со скрытием вопроса.
* Видеть список всех вопросов без ответа во всех темах в порядке их добавления. И иметь возможность их редактировать и удалять.

**Интерфейс**

* За основу пользовательского интерфейса можно взять [этот шаблон](http://netology-university.bitbucket.org/php/graduate-work/faq/resources/faq.zip) ([демо](http://netology-university.bitbucket.org/php/graduate-work/faq/resources/preview/index.html)).
* Для построения интерфейса системы администрирования можно использовать фреймворки [Bootstrap](http://getbootstrap.com/) или [Materialize](http://materializecss.com/).
* Все интерфейсные решения — на ваше усмотрения. Делайте как проще. Но основной приоритет — простота использования неподготовленным пользователем.

**Требования к организации системы**

* Система должна быть реализована на PHP.
* Использована MVC архитектура.
* Все компоненты должны быть реализованы с использованием ООП.
* В качестве шаблонизатора использовать [Twig](http://twig.sensiolabs.org/). Если вы предпочитаете другой шаблонизатор — выбор нужно будет обосновать.
* Данные администраторов, тем, вопросов и ответов должны хранится в СУБД MySQL.
* Система при работе не должна вызывать исключений и ошибок, в том числе уровня E\_WARNING и E\_STRICT.

**Что необходимо предоставить по проекту**

* Описание системы. Ссылка на Google-документ, в котором описано как устроена ваша система и нарисована UML-схема базы данных.
* Инструкции по установке и первому запуску. Файл README.md или README.html в папке проекта.
* Структуру базы данных в файле faq.sql.
* Все необходимые файлы для работы системы, а так же README.md и faq.sql должны быть доступны в репозитории на [BitBucket](https://bitbucket.org/) или [GitHub](https://github.com/), или в рабочем пространстве в [Cloud9](https://c9.io/). Нужно предоставить доступ к этим системам.
* Актуальная версия проекта должен быть опубликован на university.netology.ru и необходимо предоставить ссылку.

**Дополнительные задачи**

* Действия администраторов должны журналироваться в файл. Примерно в таком виде:

[2016-05-01 12:00:00] admin создал тему "Вопросы по PHP" (12)

[2016-05-01 13:00:00] admin удалил вопрос (99) из темы "Вопросы по PHP" (12)

[2016-05-01 13:07:00] admin обновил вопрос (100) из темы "Вопросы по PHP" (12)

[2016-05-01 13:10:00] admin опубликовал вопрос (100) из темы "Вопросы по PHP" (12)

* Реализовать блокировку вопросов по ключевым словам. Администратор ведет справочник запрещенных слов. И если в вопросе встречается запрещенное слово, то вопрос помечается как заблокированный. Такие вопросы видно только в отдельном списке «заблокированные вопросы». Там их можно просмотреть и разблокировать. Должно быть видно по каким словам вопрос был заблокирован.
* Реализовать [Телеграм-бота](https://telegram.org/), которому можно задать вопрос. Вопрос добавляется в общий список. Когда администратор опубликует ответ на вопрос, то бот должен так же отправить ответ сообщением в Телеграм пользователю, который его задал.