Отчёты к лабораторным работам

по дисциплине  
«Веб-технологии»

выполнил   
студент гр. ИС/б-18-1-з Демиденко А. А.  
зачётная книжка № 481483  
принял Дрозин А. Ю.

Лабораторной работа № 8  
«Исследование архитектуры MVC приложения и возможностей обработки данных HTML-форм на стороне сервера с использованием языка PHP»

1. **Цель работы**

Исследовать особенности построения приложений с использованием архитектуры MVC. Изучить основы синтаксиса PHP и принципы функционирования MVC приложения на стороне сервера. Приобрести практические навыки использования языка PHP для генерации HTML-кода и обработки HTML-форм в MVC приложении.

1. **Постановка задачи**
2. В соответствии с рекомендациями, представленными в п. 1.2.8 реструктурировать код персонального сайта на основе шаблона MVC.
3. Для страницы «Фотоальбом» реализовать хранение данных фото (подписей и имен файлов) в соответствующих константах модели Photo. Вывод таблицы, содержащей фото, реализовать в представлении с использованием операторов циклов.
4. Для страницы «Мои интересы» реализовать хранение данных (название категорий, интересов, описание интересов) в соответствующих константах модели Interest. Вывод меню ссылок и интересов реализовать в представлении с использованием операторов циклов.
5. Реализовать класс FormValidation, выполняющий валидацию данных форм, передаваемых на сторону сервера. Рекомендуемая структура класса:

* Rules – поле(массив), содержащее набор правил для проверки валидности данных;
* Errors – поле(массив), содержащее тексты ошибок, возникших при проверке валидности данных;
* isNotEmpty(data) – метод проверки является ли значение data не пустым – возвращает сообщение об ошибке, если таковая имеется;
* isInteger(data) – метод проверки является ли значение data строковым представлением целого числа – возвращает сообщение об ошибке, если таковая имеется;
* isLess(data, value) – метод проверки является ли значение data строковым представлением целого числа и не меньшим, чем value – возвращает сообщение об ошибке, если таковая имеется;
* isGreater(data, value) – метод проверки является ли значение data строковым представлением целого числа и не большим, чем value – возвращает сообщение об ошибке, если таковая имеется;
* isEmail(data) – метод проверки является ли значение data строковым представлением email – возвращает сообщение об ошибке, если таковая имеется;
* SetRule(field\_name, validator\_name) – метод, добавляющий в массив Rules проверку для поля field\_name типа validator\_name;
* Validate(post\_array) – метод, выполняющий проверку элементов в массиве post\_array, в соответствии с правилами Rules и сохраняющий сообщения об ощибках в поле Errors;
* ShowErrors() – метод, выводящий все сообщения об ошибках из поля Errors в формате HTML.

1. С использованием разработанного класса реализовать валидацию форм «Контакт» и «Тест по дисциплине “...”».
2. Реализовать дочерний класс CustomFormValidation от класса FormValidation, дополнив его возможностью выполнения специализированной проверки формы «Тест по дисциплине» на стороне сервера.
3. Реализовать дочерний класс ResultsVerification от класса Custom- FormValidation, дополнив его возможностью проверки правильности ответов, введенных пользователем на странице "Тест по дисциплине" (реализовать проверку правильности для вопросов с элементами ввода типа RadioButton, ComboBox или однострочный текст) и вывода результатов проверки пользователю.
4. **Программный код**

Программный код доступен по ссылке:

[https://github.com/worm2fed/university/tree/master/SevGU/7 semestr/WEB/labs/lab8](https://github.com/worm2fed/university/tree/master/SevGU/7%20semestr/WEB/labs/lab8)

**Выводы**

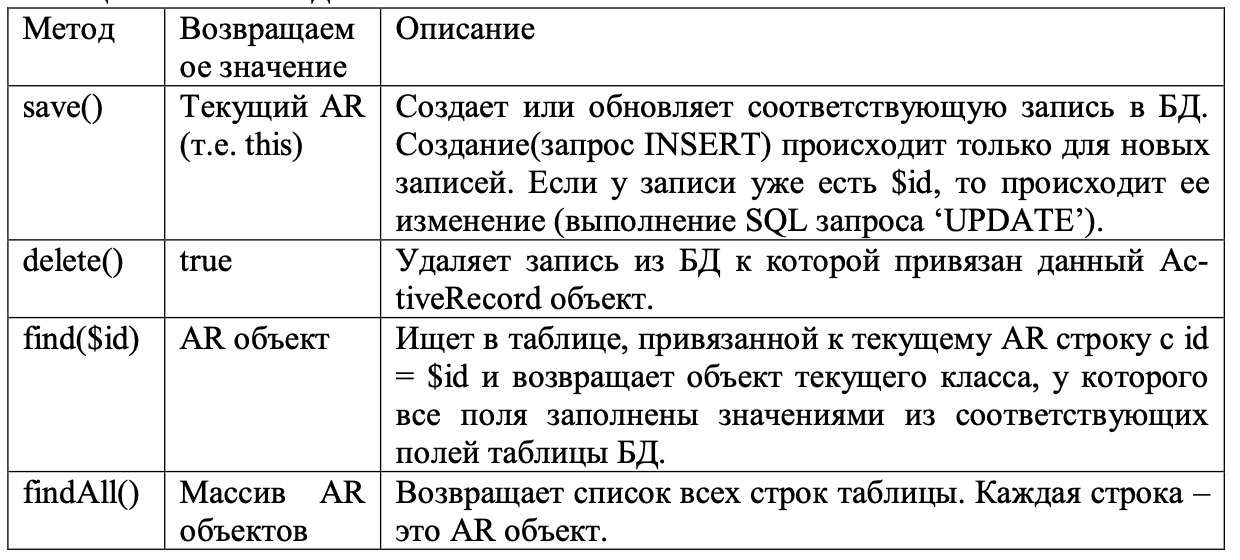
В ходе лабораторной работы были изучены возможности языка PHP и особенности архитектуры MVC, на практике закреплены базовые знания.

Лабораторной работа № 9  
«Исследование возможностей хранения данных на стороне сервера. Работа с файлами. Работа с реляционными СУБД»

1. **Цель работы**

Исследовать возможности хранения данных на стороне сервера, изучить работу c файлами и СУБД MySQL из PHP, приобрести практические навыки организации хранения данных на стороне сервера в файлах, в базах данных MySQL, а также постраничного вывода данных.

1. **Постановка задачи**
2. Разработать базовый класс BaseActiveRecord для работы с базой данных, который реализует паттерн ActiveRecord (данный класс разместить в папке /my\_site/app/core). Данный класс должен реализовать следующие функции:



1. Для всех моделей, которые будут использоваться при выполнении данной лабораторной работы, создать классы, наследующие BaseAc- tiveRecord. Для каждого из классов определить все поля и названия таблиц (данные классы необходимо разместить в папке /my\_site/app/models).
2. Создать новую страницу "Гостевая книга". Страница должна содержать форму ввода (Фамилия, Имя, Отчество, E –mail, Текст отзыва), а также таблицу сообщений, оставленных пользователями. Сообщения в таблице должны располагаться в порядке убывания даты добавления сообщения. Для хранения сообщений пользователей использовать текстовый файл messages.inc, содержащий разделенные символом «;» данные: Дату сообщения, ФИО, E–mail и Текст отзыва (для получения текущей даты сервера возможно использовать PHP функцию date('d.m.y')).
3. Реализовать страницу "Загрузка сообщений гостевой книги", содержащую форму загрузки подготовленного заранее файла messages.inc на сервер.
4. Реализовать на странице "Тест по дисциплине" сохранение ответов пользователей и правильности ответов в разработанную таблицу базы данных MySQL, с возможностью просмотра сохраненных данных (дата, ФИО, ответы, верно/неверно).
5. Разработать страницу «Редактор Блога», позволяющую добавлять записи Блога. Страница должна содержать форму добавления записи Блога и список выдаваемых постранично записей отсортированных в порядке убывания даты. Форма добавления должна содержать поля ввода:

* Тема сообщения – поле ввода однострочного текста (заполнение обязательно);
* Изображение – поле ввода файла (заполнение не обязательно);
* Текст сообщения – поле ввода многострочного текста.  
  Данные хранить в разработанной таблице базы данных MySQL. Валидацию данных осуществлять с использованием класса FormValidation, разработанного при выполнении ЛР №8.

1. Разработать страницу «Мой Блог», содержащую упорядоченные в порядке убывания даты добавления, выдаваемые постранично данные:

* Дата и время сообщения;
* Тема сообщения;
* Изображение;
* Текст сообщения.

Данные извлекать из таблицы базы данных MySQL. Формат постраничного вывода определяется как:



1. Реализовать возможность добавления записей на страницу «Мой Блог» из файла формата CSV, содержащего следующие поля: title, message, author, created\_at. Например: "тема 1","сообщение 1","Vasiliy","2019-01-01 14:00". Для этого необходимо разработать страницу «Загрузка сообщений блога», содержащую форму загрузки файла формата CSV. Добавление записей из файла в БД осуществлять с использованием подготавливаемых запросов (см. п. 2.2.4). Валидацию данных осуществлять с использованием класса FormValidation, разработанного при выполнении ЛР №8.

**3. Программный код**

Программный код доступен по ссылке:

[https://github.com/worm2fed/university/tree/master/SevGU/7 semestr/WEB/labs/lab9](https://github.com/worm2fed/university/tree/master/SevGU/7%20semestr/WEB/labs/lab9)

**Выводы**

В ходе лабораторной работы были изучены возможности хранения данных на стороне сервера, изучена работа c файлами и СУБД MySQL из PHP, приобретены практические навыки организации хранения данных на стороне сервера в файлах, в базах данных MySQL, а также постраничного вывода данных.

Лабораторной работа № 10  
«Исследование механизма сессий в PHP»

1. **Цель работы**

Исследовать механизмы сохранения данных в процессе работы пользователя с сайтом, изучить возможности авторизации пользователей с использованием механизма сессий PHP, приобрести практические навыки использования переменных, сохраняющих свое значение при переходе от одной страницы сайта к другой.

1. **Постановка задачи**

1. В соответствии с рекомендациями п. 3.2.7 реализовать зону администрирования сайта и переместить в нее страницы «Редактор блога» и «Загрузка сообщений гостевой книги».

2. Реализовать накопление информации о посетителях (см п. 3.2.6) страниц пользовательского раздела сайта в разработанной для этого таблице базы данных. Структура информации:

* Дата и время посещения;
* Web-страница посещения;
* IP-адрес компьютера посетителя;
* Имя хоста компьютера посетителя;
* Название браузера, который использовал посетитель.

3. Добавить в зону администрирования сайта страницу «Статистика посещений», отображающую информацию о посетителях сайта. Реализовать постраничное отображение информации в порядке убывания даты и времени посещения.

4. Добавить в зону администратора форму входа, содержащую поля логин и пароль. Логин и пароль админстратора хранить к коде программы. Проверку пароля осуществлять сравнением зашифрованного пароля с вычисленным заранее хешем верного пароля.

5. На всех страницах зоны администратора реализовать проверку авторизации и в случае, если посетитель не авторизован как администратор –перенаправлять пользователя на страницу авторизации). Для этой цели возможно добавить в базовый контроллер раздела администратора метод аутентификации, который будет проверять установленную контроллером переменную сессии isAdmin.

6. В пользовательском разделе сайта разработать страницу «Регистрация пользователя», содержащую форму для ввода (все поля обязательны к заполнению):

* ФИО пользователя;
* e-mail пользователя;
* логин пользователя;
* пароль пользователя.

Добавлять введенную пользователем информацию в разработанную таблицу базы данных. Предусмотреть проверку на невозможность добавления двух одинаковых логинов.

7. Добавить в меню пользовательского раздела сайта ссылку «Войти» на форму авторизации. Форма должна содержать строку ввода логина, строку ввода пароля, кнопку "Войти", а также ссылку «Регистрация» (на страницу «Регистрация пользователя»).

После успешной авторизации в меню сайта вместо ссылки «Войти» должна отображаться ссылка "Выйти", при нажатии которой происходит выход авторизированного пользователя (завершение текущей сессии).

После успешной авторизации на каждой странице сайта (возможно в области меню) должно отображаться "Пользователь: ФИО" (ФИО – Фамилия Имя Отчество пользователя).

8. Реализовать отображение результатов тестирования на странице «Тест по дисциплине» только авторизованным пользователям).

1. **Программный код**

Программный код доступен по ссылке:

[https://github.com/worm2fed/university/tree/master/SevGU/7 semestr/WEB/labs/lab10](https://github.com/worm2fed/university/tree/master/SevGU/7%20semestr/WEB/labs/lab10)

**Выводы**

В ходе лабораторной работы была изучены механизмы сохранения данных в процессе работы пользователя с сайтом, изучена возможности авторизации пользователей с использованием механизма сессий PHP, приобретены практические навыки использования переменных, сохраняющих свое значение при переходе от одной страницы сайта к другой

Лабораторной работа № 11  
«Исследование возможностей асинхронного взаимодействия с сервером. Технология AJAX»

1. **Цель работы**

Исследовать возможности асинхронного обмена данными с сервером, изучить принципы асинхронной работы с сервером, приобрести практические навыки использования технологии AJAX для организации отправки данных на сервер и получения данных без полной перезагрузки страницы.

1. **Постановка задачи**



1. На странице «Регистрация пользователя» реализовать проверку занятости логина без перезагрузки страницы (использовать событие onBlur или кнопку «Проверить занятость»). Использовать для отправки данных на сервер метод и формат данных, указанный в колонке «Задание No1» таблицы 4.3 в соответствии с вариантом задания.

2. Реализовать возможность комментирования записей блога для авторизированных пользователей сайта. В случае, если пользователь авторизирован отображать после каждой записи блога кнопку или ссылку «Добавить комментарий». При нажатии кнопки/ссылки отображать модальное окно (реализовать с использванием, например, с использованием абсолютно спозиционированного блока div), содержащее поле ввода текста комментария и кнопку «Отправить». Реализовать отправку комментария на сервер и его отображение (дата и время, автор, текст комментария) под записью блога (или под уже существующими комментариями данной записи) без полной перезагрузки страницы. Комментарии хранить в специально разработанной таблице базы данных. Использовать для отправки данных на сервер метод и формат данных, указанный в колонке «Задание No2» таблицы 4.3 в соответствии с вариантом задания.

3. В зоне администратора реализовать возможность редактирования записей блога. Отображать возле каждой записи блога кнопку или ссылку «Изменить». При нажатии кнопки/ссылки отображать окно (div), содержащее поле ввода с темой записи блога и поле ввода с текстом записи блога для редактирования, а также кнопку/ссылку «Сохранить изменения». Реализовать отправку данных и их отображение без полной перезагрузки страницы. Использовать для отправки данных на сервер метод и формат данных, указанный в колонке «Задание No3» таблицы 4.3 в соответствии с вариантом задания.

1. **Программный код**

Программный код доступен по ссылке:

[https://github.com/worm2fed/university/tree/master/SevGU/7 semestr/WEB/labs/lab12](https://github.com/worm2fed/university/tree/master/SevGU/7%20semestr/WEB/labs/lab12)

**Выводы**

В ходе лабораторной работы была изучена возможность асинхронного обмена данными с сервером, изучены принципы асинхронной работы с сервером, приобретены практические навыки использования технологии AJAX для организации отправки данных на сервер и получения данных без полной перезагрузки страницы.

.

Лабораторной работа № 12  
«Исследование возможностей фреймворка Yii2 для разработки веб-приложений»

1. **Цель работы**

Ознакомиться с возможностями фреймворка Yii2 и приобрести практические навыки реализации веб-приложений с его помощью.

1. **Постановка задачи**

В ходе выполнения лабораторной работы необходимо с использованием фреймворка Yii2 реализовать страницы «Редактор Блога» и «Загрузка сообщений блога» в разделе администратора персонального сайта и «Мой Блог» в пользовательском разделе.

Предусмотреть возможность редактирования и удаления записей блога на странице «Редактор Блога», реализовав валидацию средствами Yii2.

Для этого разработать: миграцию, модель, маршруты роутинга, контроллеры, валидации, макеты пользовательской и административной части сайта, а также необходимые представления на основе имеющихся файлов (дизайн всего сайта и разработанного «Блога» должен остаться прежним).

1. **Программный код**

Программный код доступен по ссылке:

[https://github.com/worm2fed/university/tree/master/SevGU/7 semestr/WEB/labs/lab12](https://github.com/worm2fed/university/tree/master/SevGU/7%20semestr/WEB/labs/lab12)

**Выводы**

В ходе лабораторной работы было получено представление о возможностях фреймворка Yii2 и приобретены практические навыки реализации веб-приложений с его помощью.