Титульный лист

МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ

Государственное бюджетное профессиональное образовательное учреждение Московской области «Ногинский колледж»

Курсовой проект

по МДК.09.01 Проектирование и разработка веб-приложений

ПМ.09 Проектирование, разработка и оптимизация веб-приложений

Тема:

ИСПОЛЬЗОВАНИЕ ФРЕЙМВОРКА YII2 В РАЗРАБОТКЕ ВЕБ-ПРИЛОЖЕНИЯ «Служба доставки еды «Картошка у Антошки»

Разработчик: студент группы 3ИСПР2

Косяков Н.Н

(подпись)

Оценка защиты курсового проекта

« »

Дата защиты

« » 2024

Руководитель проекта:

преподаватель Степанов С.О.

(подпись)

Ногинск, 2024 г

Содержание

Оглавление

[Основная часть 5](#_Toc169384106)

[1. инсталляция всех необходимых программ и Фреймворка, пред проектная подготовка к работе; 5](#_Toc169384107)

[2. Разработка сайта. 6](#_Toc169384108)

[2.1 Html и Scss 6](#_Toc169384109)

[2.2 PHP(yii2). 8](#_Toc169384110)

[2.3 JavaScript (JQuery) 19](#_Toc169384112)

Введение

Выполнение курсового проекта – важный этап подготовки квалифицированного специалиста; оно способствует развитию навыков самостоятельной исследовательской работы, умения творчески работать с литературой. В ходе выполнения курсового проекта преследуются задачи систематизации, закрепления и расширения профессиональных знаний, применения этих знаний и навыков при разработке исследуемых вопросов и проблем.

Цель курсового проекта: Главная цель курсового проекта изучить новый backend и framework, выполнить задачи, выданные в методических указаниях и закрепить полученные знания на практике.

Задачи курсового проекта:

1.Выбрать тему проекта.

2.Изучить YII2 и jQuery + jQuery UI.

3.Установить и настроить среду для разработки.

4.Написать Веб-приложение с функционалом.

5.Написать и защитить курсовой проект.

Актуальность: Курсовая работа актуальна тем что, Сейчас все больше людей предпочитают заказывать еду через мобильные приложения или веб-сайты, что делает разработку подобного сервиса очень востребованной и перспективной. Такие приложения упрощают процесс заказа еды, делают его быстрым и удобным для пользователей.

На сегодняшний день рынок доставки еды представлен большим количеством различных компаний и платформ, предлагающих услуги доставки еды прямо к двери потребителя. Такие сервисы как Delivery Club, Яндекс.Еда, Uber Eats, и т.д. стали популярными и широко используются людьми по всему миру.

Ситуация на рынке доставки еды динамично развивается: новые игроки появляются, существующие компании улучшают свои сервисы, расширяют географию обслуживания, предлагают скидки и акции для привлечения клиентов. Конкуренция в данной отрасли остается высокой, что стимулирует компании к постоянному совершенствованию своих услуг и повышению качества обслуживания.

Таким образом, рынок доставки еды находится в стадии активного развития, предлагая большие возможности для креативных и инновационных подходов в области технологий и обслуживания клиентов.

Основная часть

1. инсталляция всех необходимых программ и Фреймворка, пред проектная подготовка к работе;

Первым что я скачал для разработки своего веб-приложения была программа PhpStorm.Phpstorm-это PhpStorm предоставляет богатый и интеллектуальный редактор кода для PHP с подсветкой кода, расширенной конфигурацией форматирования кода, проверкой на наличие ошибок на лету и умным автодополнением.

Дальше для удобной работы с PhpStorm необходимо было установить Composer. Composer-это Composer — это пакетный менеджер уровня приложений для языка программирования PHP, который предоставляет средства по управлению зависимостями в PHP-приложении. Composer разработали и продолжают поддерживать два программиста Nils Adermann и Jordi Boggiano.

Для начала работы с самим проектом необходимо было запустить Framework Yii2.Yii2 это Yii — объектно-ориентированный компонентный фреймворк, реализующий парадигму MVC. В китайском языке Yii означает «простой и прогрессивный», также Yii представляет собой акроним от «Yes It Is!»

Чтоб начать проект на Yii2 я ввел команду composer create-project --prefer-dist yiisoft/yii2-app-basic basic в консоль PhpStorm.

После начала работы с проектом появилась необходимость сделать оформление сайта для этого я использовал scss и jQuery. Scss это метаязык на основе CSS, предназначенный для увеличения уровня абстракции CSS-кода и упрощения файлов каскадных таблиц стилей. Включён в состав языка разметки Haml. Имеет два синтаксиса: sass — отличается отсутствием фигурных скобок, в нём вложенные элементы реализованы с помощью отступов Для работы с Scss необходимо было установить Node.js Node или Node.js — программная платформа, основанная на движке V8, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. .JQuery-это набор функций JavaScript, фокусирующийся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX. Для установки jQuery необходимо с официального сайта скачать сжатый файл с кодом библиотеки, данный файл я поместил в папку js. После чего для работы jQuery в тег head нужно поместить <script defer src="js/jquery-3.6.3.js"></script>.

1. Разработка сайта.
   1. Html и Scss

В первую очередь в основу разработки сайта легли такие языки как html и css.Первым что я начал делать с сайтом это начал писать header,в основу хедера вошли блок с названием сайта,список для навигации и блоки с кнопками входа и корзины.Для написания блока мне понадобилось смотреть примеры с других сайтов чтоб примерно понимать как лучше расставлять блоки по месту нахождению, написав примерный код для структуры хедера я начал делать scss и писать дизайн для этого блока я задал темный цвет блока для заднего фона хедера и с помощью display flex расставили блоки в правильном порядке,цвет текста для блоков я выбрал белым потому что по моему мнению белый цвет лучше всего подходит к темному оформлению сайта.

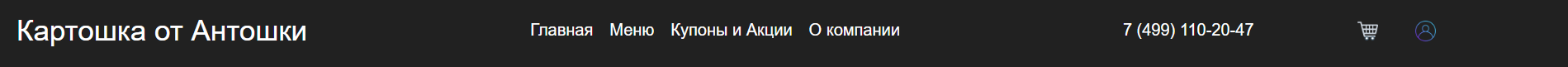


Рис.1.Header

Второй секцией я сделал секцию промо в которую вошли преимущества данного заведения и кнопка перехода на меню сделал я это в 2 разных блоках в 1 секции для данного блока я также сделал дизайн в первую очередь добавив фотку заднего фона и сделав определенный разграниченный размер для данной секции.Также я поменял цвет текста у блока главный слоган я оставил белым,а уже остальные причины я решил сделать бледно желтым цветом на мой взгляд показалось что этот цвет хорошо сочетается с задним фоном блока.

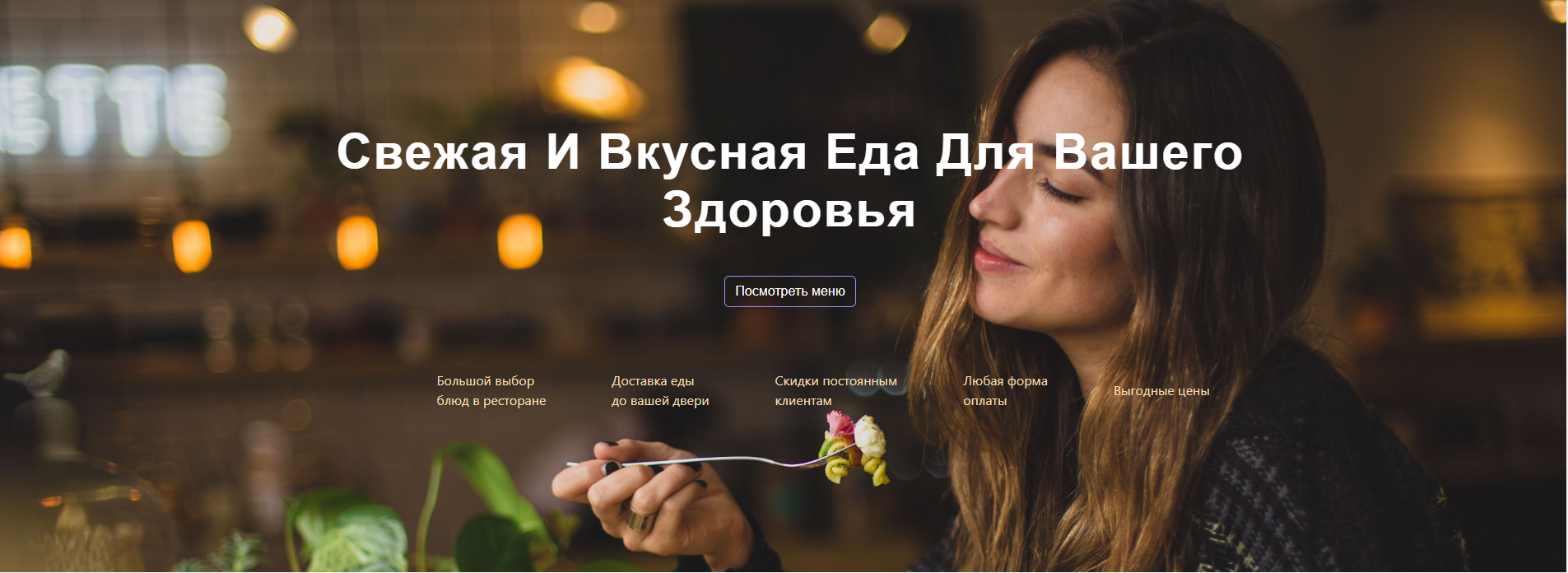


Рис.2.Вид секции промо.

Третий секцией я решил сделать секцию с заказами блюд на мероприятия для этого были сделаны 2 блока с 2 мероприятия;

1.Создания персонального блока с меню на свадьбу.

2.Создания персонального блока с меню на день рождения.

В основу дизайна этого блока вошло создание карточки для этих мероприятий. Для который я задал цвета блока белым,а названия услуги я решил сделать желтым чтоб отличался от описания услуги.

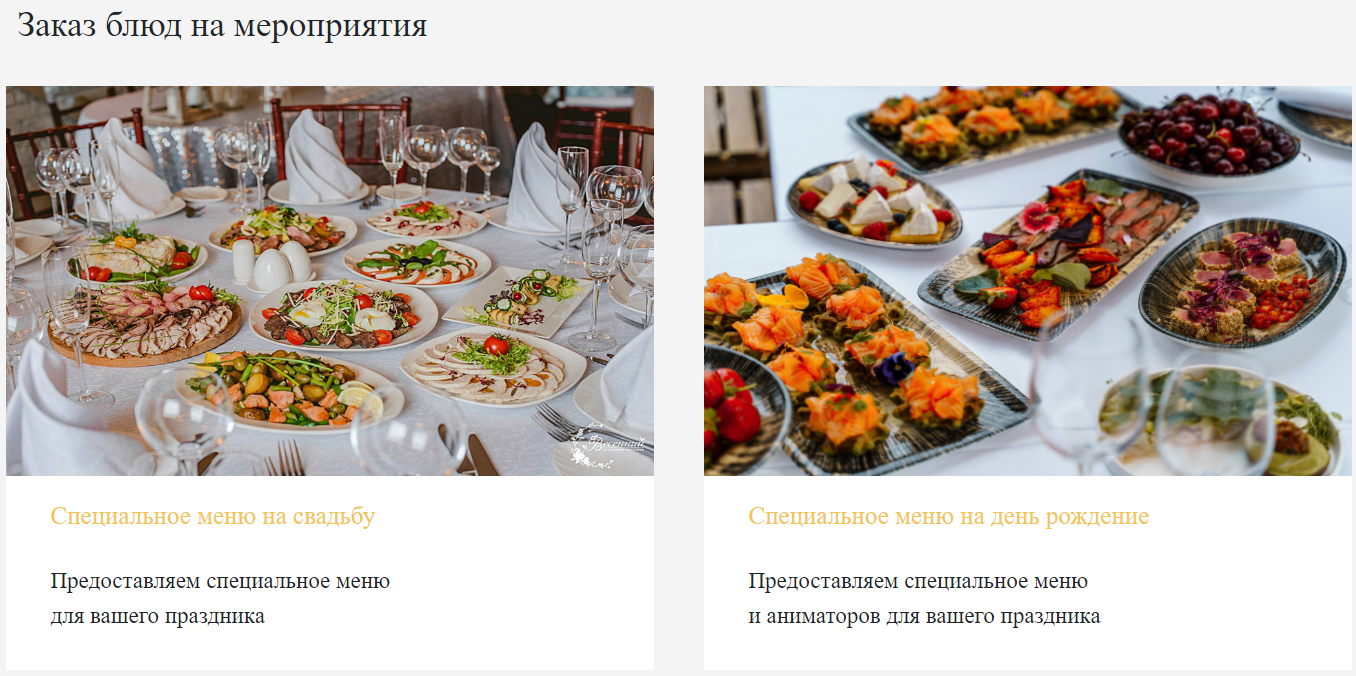


Рис.3.Вид блока заказа меню для мероприятия

Четвертая секция это секция с меню и сортировкой в первую очередь я сделал кнопки для сортировки по категориям товаров, Был сделан заголовок для всего меню и для подкатегорий с еженедельными и ежедневными акциями и добавлены кнопки названия для того чтоб было легче выбрать что именно ты хочешь заказать. дальше я сделал заголовок и кнопку смотреть всё я поместил в линию сделав небольшой отступ между ними, дальше в ряд стоят кнопки категорий которые я закруглил, сделал чёрную обводку а задний фон белый в центр кнопок поставил текст и уже ниже отдельным рядом сделал кнопки сортировки подобно категориям и так же я заранее прописал для них выдвижение при наведении но сделал функционал этого позже через jquery.



Рис.4 Блок сортировки меню

Пятой секцией стал блок о нас в который я добавил описание кафе и фотографию повара и официанта заведения. также в эту секцию вошла фотография подписи владельца и сделано так чтоб данная подпись гармонично смотрелась в блоке.стиль я решил сделать таким образом чтоб фотографии с официантом и поваров накладывались друг на друга,а текст с подписью был отдельно, сделано это было через flexы и на мой взгляд это получилось очень гармонично.



Рис.5.блок о нас.

Последней секцией стал footer который я делал также, брав пример с других сайтов. В основу футера я решил добавить адрес заведения, номер и пункт того что все права защищены. В дизайне я в первую очередь разместил 3 разных блока с информацией в 3 разные части хедера с помощью flexов так же я решил задать белый цвет текста и темный цвет фона чтоб это также гармонично смотрелась на сайте.

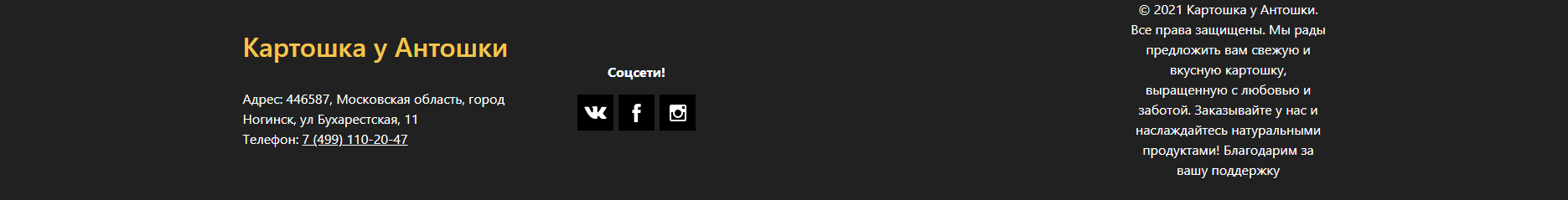


Рис.6.Footer.

* 1. PHP(yii2).

Дальше для работы сайтом необходимо было начать работать с php, первым что я сделал для работы с ним это создал базу данных и попытался вывести ее разными способами спустя какое-то время я нашел способ для вывода и вывел ее с помощью функции PDO для этого я написал код, который вызывает базу данных по ее названию и выводит данные о товаре по вызываемым столбцам выглядело это примерно так:

<?php  
$pdo = new PDO('mysql:host=localhost;dbname=yii2basic', 'root', '');  
  
// Получение данных о товарах из базы  
$statement = $pdo->query('SELECT \* FROM TOVAR');  
$tovars = $statement->fetchAll(PDO::*FETCH\_ASSOC*);  
?>

Далее я начал создавать оформление с выводом данных из таблицы что б получилась карточка товара, сделано это было с помощью, переменной которую я создал в функции PDO и названия столбцов. Также было сделана кнопка для добавления товара в корзину.



Рис.7.Код карточки товара.

Добавление товара в корзину является важной частью любого заведения поэтому. Для создания корзины необходимо было написать код для модели, контроллера и представления. Сначала я создал модель cart в который в 1 очередь вызвал таблицу TOVAR для того чтоб функция добавления могла понимать, что именно я хочу добавить в корзину, также я задал функцию Add item которая используй таблички и типы данных и делает так чтоб при каждом нажатии на кнопку добавить в корзину проверялось наличие данного столбца в таблице.

После этого я перешел на добавление кода функции в контролере 1 функцией стало добавление товара в корзину, в которой в 1 очередь основным берется id, при совпадении которого при каждом нажатии на данный товар увеличивается количество товара в корзине. Следующей функцией была сделана функция удаления из корзины, которая проверяет id товара и удаляет полностью все количество данного товара из корзины.

Следующем этапом была сделана функция обновления количества товара, сделано это для того чтоб если человек хочет увеличить или уменьшить количество данного товара он мог просто написать количество, которое ему хочется, а не нажимать много раз на кнопку.

Дальше была сделана функция для подсчета общей цены и сделано было это так что цена товара умножалась на количество и выводилось на основе этих данных.

public function removeItem($id)  
 {  
 if (isset($this->items[$id])) {  
 unset($this->items[$id]);  
 }  
 }  
  
 public function updateItem($id, $quantity)  
 {  
 if (isset($this->items[$id])) {  
 $this->items[$id]['quantity'] = $quantity;  
 }  
 }  
  
  
 public function getTotal()  
 {  
 $total = 0;  
 foreach ($this->items as $item) {  
 // Проверка типа данных перед выполнением операции  
 if (is\_numeric($item['price']) && is\_numeric($item['quantity'])) {  
 $total += $item['price'] \* $item['quantity'];  
 } }  
 return $total;  
 }  
}

Следующим необходимо было сделать страницу представления для корзины, В 1 очередь я сделал наименования столбцов чтоб каждый блок товара был под своим названием было сделано это для удобного просматривания своего заказа также если корзина пустая выводится сообщение о том, что корзина пуста. При добавлении товара в корзину было сделано так чтоб выводилась не вся информация о товаре, а только основную информацию это фотография, наименование и цена, так было сделано для того чтоб не засорять таблицу ненужной информацией о товаре и спокойно прочитать все что нужно и уже спокойно оформить заказ, только при оформлении заказа необходимо было войти в аккаунт чтоб при успешном оформление заказа ты мог посмотреть свой заказ через личный кабинет.

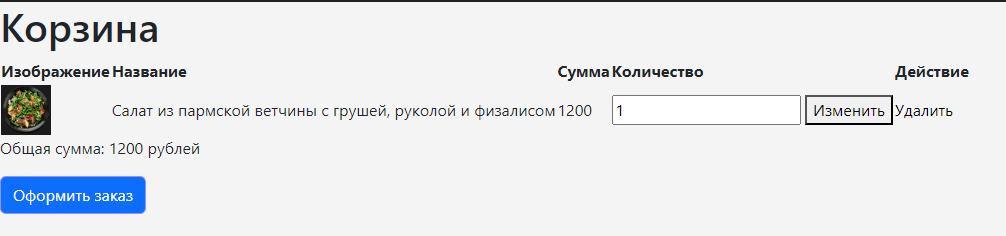


Рис.8.Вид корзины с товаром.

Далее необходимо было сделать функцию авторизации и регистрации для этого была создана таблица user, также необходимо было использовать Модели, контролер и страницы представления. В таблице User находятся такие столбцы как Username, password, ключ аутоинфекции, и роль, которая делилась на 3 разные роли;

1. Admin.

2. Courier.

3. User.

Следующим этапом я создал модель user уже в которой я начал прописывать правила для идентификации пользователя и его данных таких как;

1. Username.
2. Password.
3. Ключ идентификации.
4. Email

А также для того чтоб скрывался пароль в базе данных. Далее я создал модель регистрации где были вызваны атрибуты для имени, пароля. Затем создал функцию, регистрация которая создавала цикл if для того чтоб в будущем можно было использовать поля для ввода информации при регистрации такой как;

1. Username.
2. Password.
3. email

В случае если все данные введены успешно попадали в таблицу и аккаунт успешно регистрировался в случае если какое-то поле не было заполнено выдавалась ошибка о том, что необходимо обязательно заполнить данное поле.

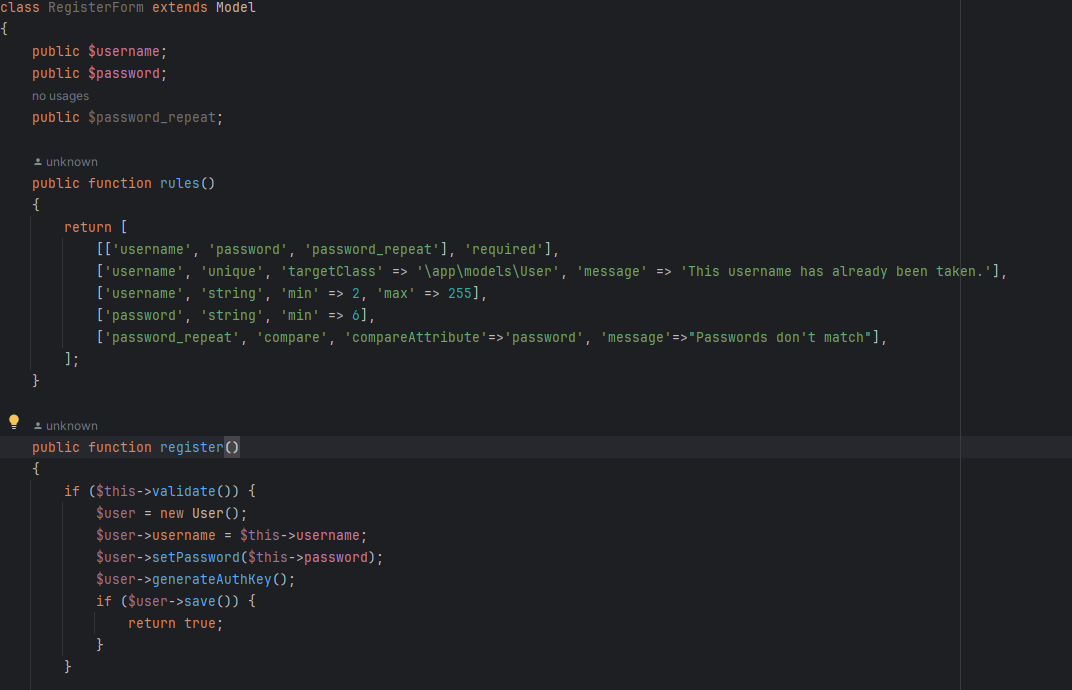


Рис.9.Модель регистрации.

Для регистрации была создана отдельная функция, в контролере которая создает нового пользователя и при успешной регистрации перенаправляет на страницу авторизации. Для страницы представления был создан файл signup который отвечал за вид данной страницы. В нем были созданы поля для ввода Usermame, email и пароль в случае если поле не заполнено выдавалась ошибка. После была сделана кнопка зарегистрироваться которая уже осуществляет функцию успешной регистрации пользователя.

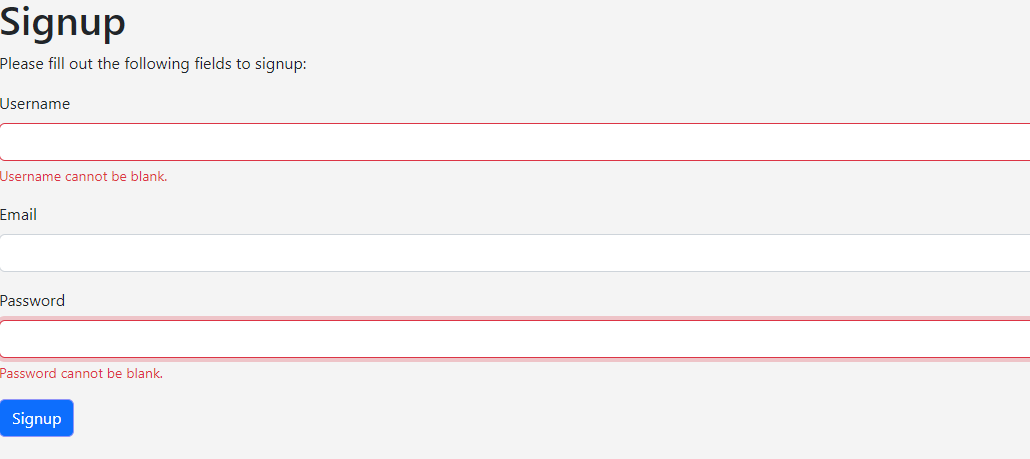


Рис.10.Представление страницы регистрации

Далее необходимо было сделать модель логина в которой вызваны атрибуты username и password, галочки запомнить меня. Написал функцию правил для атрибутов так чтоб пароль проходил валидацию, для этого я сделал функцию, для пароля которая проверяет данные из таблицы user и если все правильно, то не выдает ошибок, а если есть ошибка, то пишет, что пользователь не найден. И также была создана функция логина, которая позволяет пользователю успешно войти в аккаунт. Для логина в контролере была создана функция, которая создает форму, в которой если пользователь зашел в аккаунт с определённой роли, то для каждой роли открывается своя страница. В представлении был создан файл login в которой был написан заголовок и форма для заполнения атрибуты которой брались из модели, также была создана кнопка войти и кнопка перехода на регистрацию если у вас еще нету аккаунта на данном сайте.

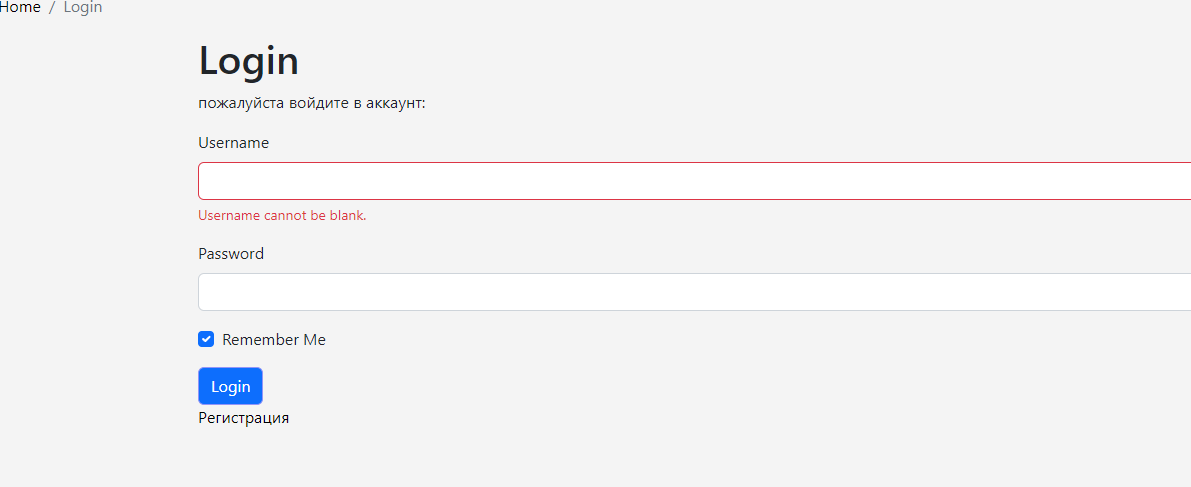


Рис.11.Страница авторизации.

Следующим необходимо было создать страницу для оформления заказа для чего были созданы функции регистрации и авторизации. Для начала я создал таблицу orders в который использовались столбцы такие как;

1. id.

2. id пользователя (связанное с таблицей user).

3. Customer name.

4. Customer email.

5. Address.

6. Phone number.

7. Delivery method.

8. Promo code.

9. Status.

10. Дата создания заказа.

11. Дата изменения заказа.

После я написал модель order в которой сделал константы для всех столбцов, которые необходимо будет заполнить в форме заказа, далее я вызвал таблицу orders и создал правила для всех столбцов из таблицы. Также я создал правило для проверки промокода которые брались из модели promo code и проверялись на наличие в таблице которая была вызвана с помощью модели promo code.



Рис.12.Вид модели orders.

Далее я создал функцию, в контроллере которая проверяет id авторизированного пользователя и связывает его с заказом и после этого попадает в базу данных если заказ успешно оформлен и выдается сообщение об успешном оформление заказа. Код контролера выглядит так

public function actionCheckout()  
{  
 $model = new Order();  
  
 // Устанавливаем user\_id текущего пользователя, если он авторизован  
 if (!Yii::*$app*->user->isGuest) {  
 $model->user\_id = Yii::*$app*->user->id;  
 }  
  
 if ($model->load(Yii::*$app*->request->post()) && $model->save()) {  
 Yii::*$app*->session->setFlash('success', 'Заказ успешно оформлен!');  
 return $this->refresh();  
 }  
  
 return $this->render('checkout', [  
 'model' => $model,  
 ]);  
}

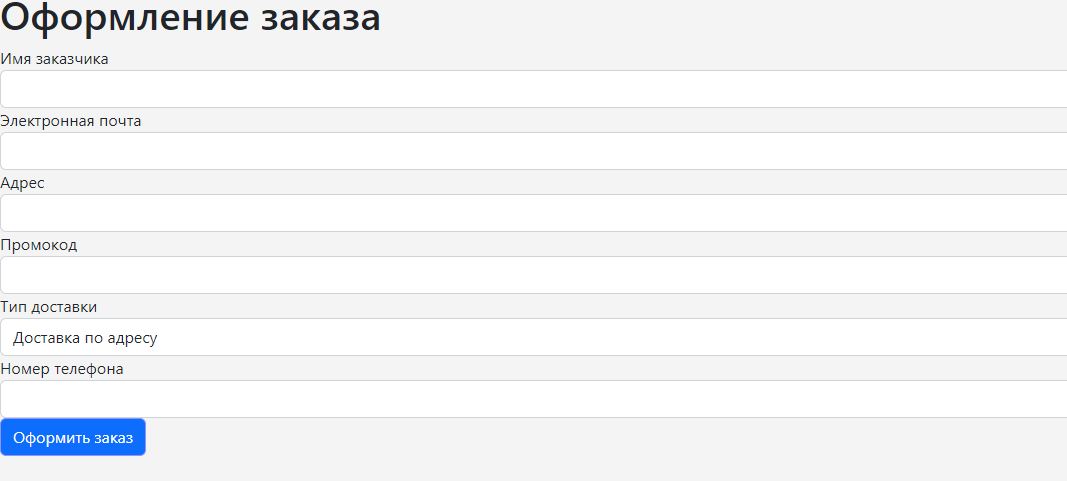


Рис.13.Вид страницы оформления заказа.

Далее я решил создать личную страницу пользователя для этого также были созданы контролер и страница представления. В контроллере проверялось связан ли id пользователя с таблицей если такой id есть, то успешном открывалась страница пользователя в которой выводились данные о всех заказах, которые пользователь делал под данным аккаунтом, а также было выведено имя пользователя и почта которые он указал пре регистрации. страница представления была создана таким образом, что сначала выводились данные пользователя, а уже потом данные о заказах авторизированного пользователя. Для нее уже есть таблица order которая была создана для оформления заказа и через нее идет подключение к базе данных. Необходимо было создать функцию для страницы профиля в которой я написал, что при заходе на страницу профиля проверяется id пользователя и если его нету, то выдается ошибка ‘не авторизован’, также он проверяет id пользователя для того чтоб функция могла выдать все заказы, связанные с данным пользователем. Затем я начал писать код для страницы представления в ней были вызваны данные из таблицы order такие как;

1. id заказа.
2. Имя заказчика.
3. Адрес доставки.
4. Статус заказа.

В случае если заказов не находилось, то выдавалось сообщение о том, что заказов не найдено.

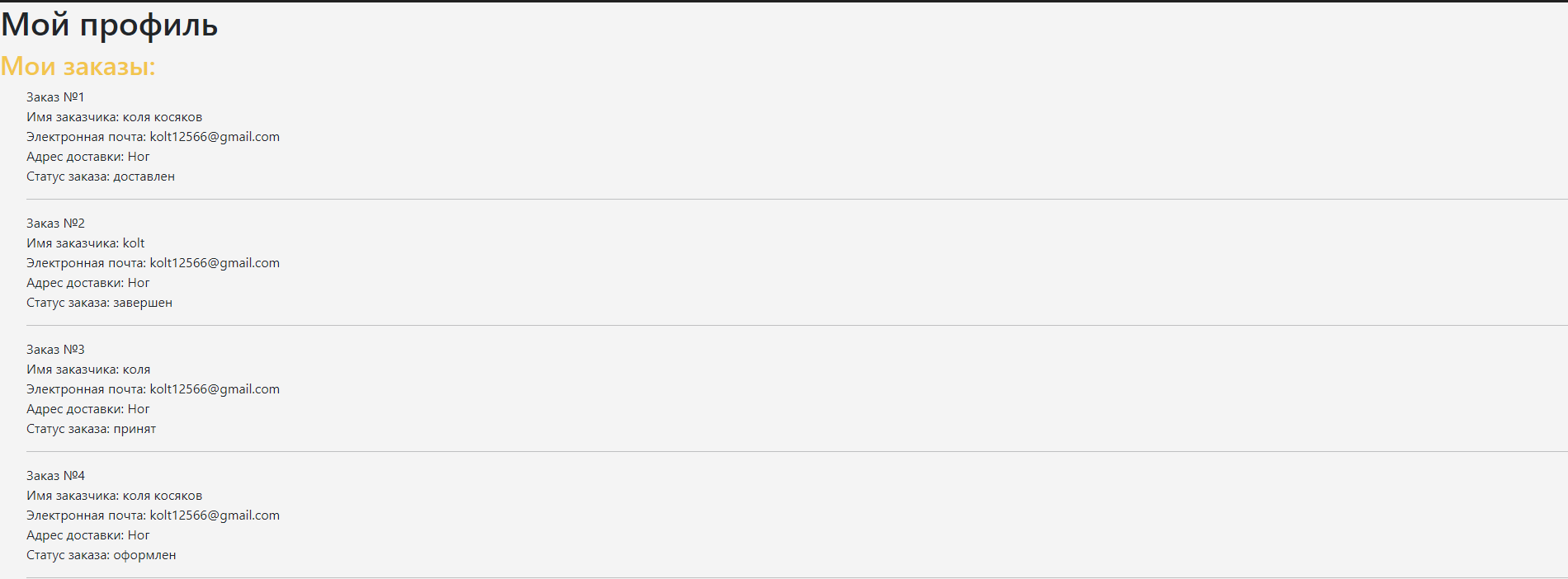


Рис.14 Вид страницы заказов.

Предпоследним что я решил сделать это была страница курьера в которой можно было менять статус заказа для нее мне было необходимо было создать контролер и страницу представления. В контроллере проверялся order id, и есть ли такая модель и столбец чтоб можно было менять статус заказа, если все успешно проверялось, то открывалась страница пользователя в которой выводились данные о всех заказах, которые пользователи делали под авторизированными аккаунтами. страница представления была создана таким образом, что сначала выводились данные о заказе и поле для смены статуса заказа, а уже потом кнопка подтвердить изменения если успешно получалось изменить статус то выдавалось сообщение 'Статус заказа успешно обновлен.' А в случае если не получалось поменять статус, то 'Ошибка при изменении статуса заказа.'

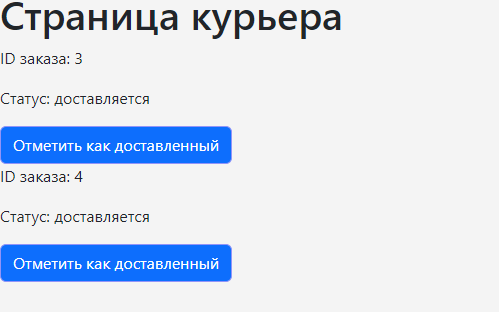


Рис.15.вид страницы курьера.

А также для навигации была прописана команда для кнопки перехода на курьера которая видна только пользователю с ролью Courier

Последним что я сделал это была панель администратора для которой мне нужно было сделать 5 задач, первая эта возможность работать с таблицей товаров, вторая эта возможность работать с таблицей акций, третья эта возможность работать с таблицей промокодов, четвертая задача эта возможность изменять роль у юзеров, пятая это возможность изменять статус заказа у пользователей. Для первой задачи мне понадобилась модель Tovar, в контролере я создал функции создания, обновления и удаления, которые имеют тот же принцип действия и похожий код который был использован для корзины. После я создал представление для панели администратора где сначала идёт заголовок затем кнопки для перехода к страничке следующих задач. Затем открыл php где с помощью цикла if я прописал кнопку для создания нового товара с использованием функции create из контроллера, а потом написал какие колонки должны выводится, а это все из таблицы Tovar. Так же прописал кнопки для обновления и удаления, которые используют функции из контроллера, затем прописал else с циклом if где если пользователь нажимает создать или обновить товар, то у него открывается формы для заполнения всех столбцов из таблицы, так же для категорий прописал какие именно значения берутся из таблицы и написал им тоже самое название. Дальше создал кнопку, которая отвечает либо за создание, либо за обновление товара.

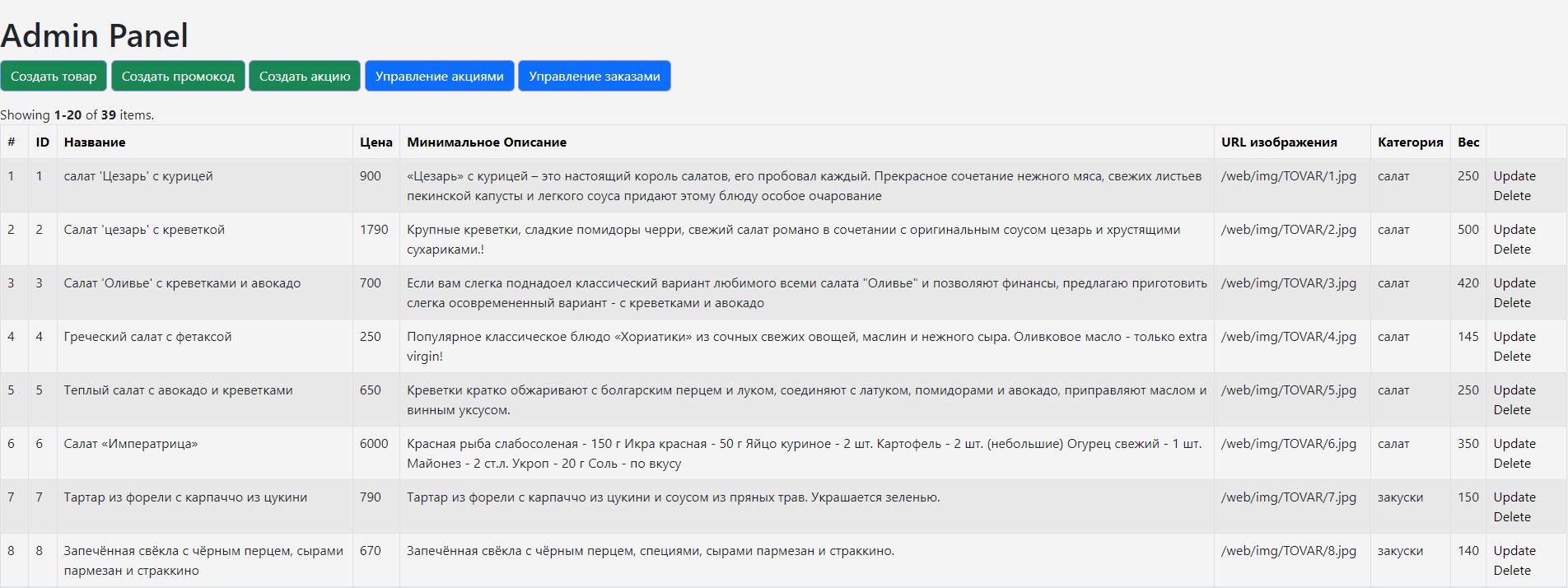


Рис.16.Вид страницы админа для товаров

А также для навигации была прописана команда для кнопки перехода на админку которая видна только пользователю с ролью Admin

Для второй задачи мне понадобилась модель promotions, в контролере я создал функции создания, обновления и удаления, которые имеют тот же принцип действия что и в прошлой задаче. Затем открыл php где с помощью цикла if я прописал кнопку для создания новой акции с использованием функции create из контроллера, а потом написал какие колонки должны выводится, а это все из таблицы Promotions. Так же прописал кнопки для обновления и удаления, которые используют функции из контроллера, затем прописал else с циклом if где если пользователь нажимает создать или обновить промокод, то у него открывается формы для заполнения всех столбцов из таблицы, так же для типа акции прописал какие именно значения берутся из таблицы и написал им тоже самое название. Дальше создал кнопку, которая отвечает либо за создание, либо за обновление товара.

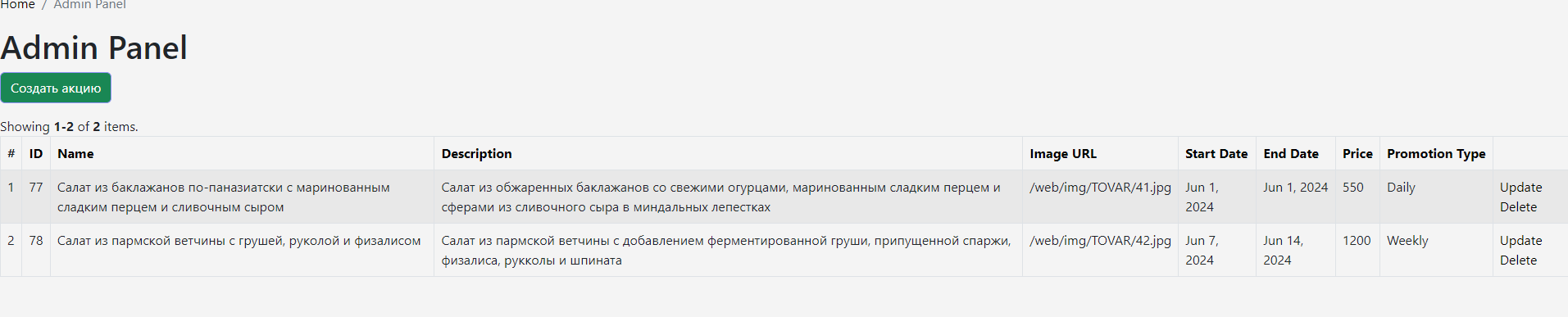


Рис.17. Вид страницы админа для акций.

Для третьей задачи мне понадобилась модель promo code, в контролере я создал функции создания, обновления и удаления, которые имеют тот же принцип действия что и в прошлой задаче. Затем открыл php где с помощью цикла if я прописал кнопку для создания новой акции с использованием функции create из контроллера, а потом написал какие колонки должны выводится, а это все из таблицы Promo code. Так же прописал кнопки для обновления и удаления, которые используют функции из контроллера, затем прописал else с циклом if где если пользователь нажимает создать или обновить промокод, то у него открывается формы для заполнения всех столбцов из таблицы. Дальше создал кнопку, которая отвечает либо за создание, либо за обновление товара.

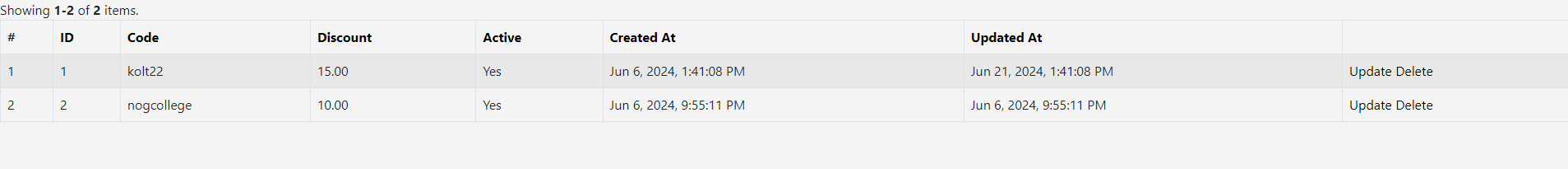


Рис.18. Вид страницы админа для акций.

Для четвертой задачи уже есть модель user, в контроллере я прописал первую функцию UpdateUser в котором берутся данные и таблички user и выводится на страничке представления. Вторая функция отвечает за изменения роли пользователя, для этого я прописал через метод post вызов id и role которые он берёт из модели user, затем через цикл if прописал все константы роли пользователя и ещё раз открыл if где при сохранении изменения выдаёт либо что всё прошло успешно, либо ошибка. Следующим я создал представление, в котором сначала идёт заголовок, потом открыл php в котором, как и для панели администратора написал вывод всех столбцов из таблицы user и добавил кнопку для перехода к обновлению роли пользователя. Затем в следующем div я написал с помощью php форму для изменения роли пользователя который использует функцию из контролера и кнопку для обновления этой роли.



Рис.19. Вид страницы админа для пользователей.

Для второй задачи уже есть модель order, в контроллере я прописал первую функцию UpdateStatus в котором берутся данные и таблички orders и выводится на страничке представления. Вторая функция отвечает за изменения статуса заказа, для этого я прописал через метод post вызов id и status которые он берёт из модели order, затем через цикл if прописал все константы статуса заказа и ещё раз открыл if где при сохранении изменения выдаёт либо что всё прошло успешно, либо ошибка, а также прописал ошибку если заказ некорректный или статус и так же вывод для представления.

Следующим я создал представление, в котором сначала идёт заголовок, потом открыл php в котором, как и для панели администратора написал вывод всех столбцов из таблицы orders и добавил кнопку для перехода к обновлению статуса заказа. Затем в следующем div я написал с помощью php форму для изменения статуса заказа который использует функцию из контролера и кнопку для обновления этого статуса.

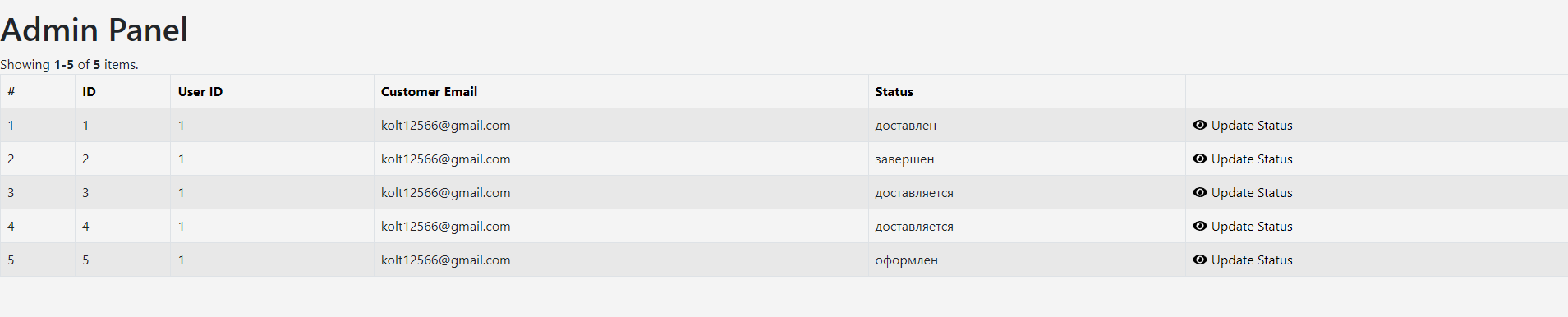


Рис.20. Вид страницы админа для статуса заказа.



2.3 JavaScript (JQuery)