ОБЛАСТНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

**«ТОМСКИЙ ЭКОНОМИКО-ПРОМЫШЛЕННЫЙ КОЛЛЕДЖ»**

*Методическое объединение информационных технологий*

**УЛЯХИН ВАСИЛИЙ АЛЕКСЕЕВИЧ**

**ПОДГОТОВКА К ДЕМОНСТРАЦИОННОМУ ЭКЗАМЕНУ**

**КОМПЕТЕНЦИЯ «ВЕБ-ТЕХНОЛОГИИ»**

*Методическое пособие*



ТОМСК 2022

ОГЛАВЛЕНИЕ

[ЗАДАНИЕ ДЕМОНСТРАЦИОННОГО ЭКЗАМЕНА ПО КОМПЛЕКТУ ОЦЕНОЧНОЙ ДОКУМЕНТАЦИИ 3](#_Toc132023551)

[КРИТЕРИИ ОЦЕНИВАНИЯ 7](#_Toc132023552)

[ПРИМЕР РЕШЕНИЯ ЗАДАНИЯ ДЭМОНСТРАЦИОННОГО ЭКЗАМЕНА 15](#_Toc132023553)

[Установка фреймворка Yii 2 15](#_Toc132023554)

[Первоначальная настройка фреймворка Yii 2 16](#_Toc132023555)

[Разработка базы данных 19](#_Toc132023556)

[GII 22](#_Toc132023557)

[Авторизация 23](#_Toc132023558)

[Регистрация 25](#_Toc132023559)

[Меню сайта 28](#_Toc132023560)

[Админ панель 29](#_Toc132023561)

[Категории товаров 30](#_Toc132023562)

[Товары 31](#_Toc132023563)

[Страница «О нас» 37](#_Toc132023564)

[Страница «Где нас найти?» 38](#_Toc132023565)

[Корзина 39](#_Toc132023566)

[Оформление заказа 43](#_Toc132023567)

[Мои заказы 44](#_Toc132023568)

# ЗАДАНИЕ ДЕМОНСТРАЦИОННОГО ЭКЗАМЕНА ПО КОМПЛЕКТУ ОЦЕНОЧНОЙ ДОКУМЕНТАЦИИ

|  |  |
| --- | --- |
| Номер компетенции | 17 |
| Название компетенции | Веб-технологии |
| Номер КОД | 1.4 |

**ВВЕДЕНИЕ**

**Технологии этого модуля:** Разработка интернет-магазина.

**Время на выполнение:** 5 часов.

К вам обратилась молодая компания «Copy Star», которая занимается продажей копировального оборудования. Заказчик хочет получить небольшой сайт для продажи своего оборудования.

Так как заказчик ограничен в деньгах, но в будущем планируется расширять ассортимент, то он просит сделать небольшую админ-панель.

Заказчик планирует дальнейшее развитие проекта, поэтому использование фреймворков будет плюсом.

**ВНИМАНИЕ! Проверяться будут только работы, загруженные на сервер!**

**ОПИСАНИЕ ПРОЕКТА И ЗАДАЧ.**

**ОБЩЕЕ ОПИСАНИЕ ПРОЕКТА**

Все пользователи системы подразделяются на три группы:

1. Администраторы;
2. Посетитель сайта;
3. Клиент.

Функционал администратора:

1. Добавление, удаление, редактирование данных о товаре;
2. Добавление, удаление категорий товаров;
3. Просмотр всех заказов и смена их статусов.

Функционал посетителя сайта:

1. Просмотр товаров и рубрик сайта.

Функционал клиента:

1. Добавление товара(-ов) в корзину;
2. Редактирование корзины;
3. Создание заказа;
4. Просмотр своих заказов.

Ваша задача – реализовать интернет-магазин.

**ОБЩИЕ ТРЕБОВАНИЯ**

Админ панель должная быть доступна по адресу http://xxxxxx/admin под логином admin и паролем admin11. Она не должна быть доступна для других пользователей.

**СПЕЦИФИЧЕСКИЕ ТРЕБОВАНИЯ К ИНТЕРНЕТ-МАГАЗИНУ**

**Рубрики сайта**

Всем пользователям доступны рубрики: "О нас", "Каталог", "Где нас найти?". Также в меню нужно сделать кнопку вход и регистрация.

**Страница «О нас»**

На странице представлен логотип компании и написан девиз компании. Также представлен слайдер ("Новинки компании") с пятью последними добавленными товарами (выводится фото товара и его название).

**Страница «Каталог»**

На странице по умолчанию представлены все товары, упорядоченные по новизне (от новых добавленных к более старым). Упорядочить товары можно: по году производства, по наименованию, по цене. Также товары можно отфильтровать по категориям (лазерные принтеры, струйные принтеры, термопринтеры). Каждая карточка товара представлена изображением, названием и ценой. При клике на товар открывается отдельная страница товара. Пользователю всегда показывает товары, которые только есть в наличии.

**Страница «Товар»**

Каждая страница содержит фото товара, наименование, цену и характеристики (страна-производитель, год выпуска, модель).

**Страница «Где нас найти?»**

На странице располагается карта и контактные данные (адрес, номер телефона, email).

**Регистрация нового клиента**

Форма регистрации нового пользователя в системе должна содержать следующие поля:

* name – обязательное поле, разрешенные символы (кириллица, пробел и тире);
* surname – обязательное поле, разрешенные символы (кириллица, пробел и тире);
* patronymic – не обязательное поле, разрешенные символы (кириллица, пробел и тире);
* login – обязательное и уникальное поле, разрешенные символы (латиница, цифры и тире);
* email – обязательное и уникальное поле, проверка на email;
* password – обязательное поле, не менее 6-ти символов;
* password\_repeat – обязательное поле, должно совпадать с полем password;
* rules - согласие с правилами регистрации.

При ошибках валидации пользователю должны выводиться сообщения без перезагрузки страницы.

**Аутентификация**

Форма авторизации пользователя в системе должна содержать следующие поля:

* login;
* password;

При ошибках валидации пользователю должны выводиться сообщения.

**Выход**

Позволяет пользователю сделать логаут, после перенаправляет на страницу "О нас".

**ФУНКЦИОНАЛ КЛИЕНТА**

**Формирование корзины**

После авторизации у пользователя появляется кнопка "В корзину" на карточках товаров в каталоге, а также на странице самого товара. Каждое нажатие добавляет 1 товар в корзину. Нельзя добавить в корзину больше товаров, чем есть в наличии (выдается сообщение об ошибке без перезагрузки страницы).

**Управление корзиной**

В корзине можно добавить или убрать единицу каждого товара (учитывая ограничения по количеству в наличии). После чего пользователь может сформировать заказ.

**Оформление заказа**

Пользователь вводит свой пароль для подтверждения заказа и нажимает кнопку "Сформировать заказ" (выдается сообщение об ошибке без перезагрузки страницы в случае неверного ввода пароля).

**Просмотр заказов**

Пользователь может просмотреть список своих заказов (в каждом есть количество товаров, его наименование и их статус), упорядоченных от новых к старым. Пользователь также имеет возможность удалить новые заказы.

**ФУНКЦИОНАЛ АДМИНИСТРАТОРА**

**Просмотр списка всех заказов**

Администратор может просматривать все заказы, фильтруя их по: новые, подтвержденные, отмененные. В списке видны таймстамп заказа, ФИО заказчика и количество заказанных товаров. По умолчанию у всех заказов статус "Новый".

**Управление заказом**

Администратор может отменить заказ, указав причину отказа или подтвердить заказ.

**Управление товарами**

Администратор опубликовывает, удаляет и редактирует товары.

**Управление категориями**

В будущем планируется расширение спектра товаров для этого нужно сделать функцию добавления и удаления категории.

**ИНСТРУКЦИЯ ДЛЯ КОНКУРСАНТА**

Разработанный интернет магазин должен быть доступен по адресу http://xxxxxx-m5.wsr.ru/, где xxxxxx - логин участника (указан на индивидуальной карточке).

**СИСТЕМА ОЦЕНКИ**

|  |  |  |
| --- | --- | --- |
| **Секция** | **Критерий** | **Сумма** |
| 1 | Организация работы и управление | 2,00 |
| 2 | Коммуникация и навыки межличностного общения | 2,00 |
| 3 | Графический дизайн | 10,00 |
| 4 | Верстка | 10,00 |
| 5 | Программирование на стороне клиента | 10,00 |
| 6 | Программирование на стороне сервера | 10,00 |
| 7 | CMS | 0,00 |
| **Всего** |  | 44,00 |

# КРИТЕРИИ ОЦЕНИВАНИЯ

| Sub Criteria ID | Sub Criteria  Name or Description | Aspect Type  O = Obj  J = Judg | Aspect - Description | Judg Score | Extra Aspect Description (Obj or Subj)  OR  Judgement Score Description (Judg only) | WSSS Section | Max Mark |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A1 | Организация работы и управления |  |  |  |  |  |  |
|  |  | O | Результаты работы представлены по пути согласно заданию |  |  | 1 | 0.25 |
|  |  | O | Данные для авторизации администратора соответствуют заданию |  | Логин: admin  Пароль: admin11 | 1 | 0.25 |
|  |  | O | Админ панель доступна по адресу из задания |  |  | 1 | 0.25 |
|  |  | J | Представленный результат соответствует требованиям задания |  |  | 1 | 1.25 |
|  |  |  |  | 0 | Работа не выполнена |  |  |
|  |  |  |  | 1 | Реализована минимально необходимая часть функционала |  |  |
|  |  |  |  | 2 | Работа соответствует заданию, но имеются некоторые несоответствия |  |  |
|  |  |  |  | 3 | Работа полностью соответствует заданию |  |  |
| A2 | Коммуникативные и межличностные навыки |  |  |  |  |  |  |
|  |  | O | Осмысленные имена функций и переменных |  | За использование транслита штраф – 0.25 | 2 | 0.5 |
|  |  | J | Качество PHP кода |  |  |  | 0.75 |
|  |  |  |  | 0 | В коде нельзя быстро разобраться |  |  |
|  |  |  |  | 1 | Осмысленное именование переменных, функций |  |  |
|  |  |  |  | 2 | Уровень 1 + Используется PHP framework или чистый PHP (с использованием классов, ООП) |  |  |
|  |  |  |  | 3 | Используется PHP framework, код организован и легко читаем |  |  |
|  |  | J | Использование комментариев в коде |  |  | 2 | 0.25 |
|  |  |  |  | 0 | Отсутствуют комментарии, созданные пользователем |  |  |
|  |  |  |  | 1 | Используются комментарии, но их недостаточно и/или комментарии некорректны |  |  |
|  |  |  |  | 2 | Осмысленные комментарии, но возможны улучшения |  |  |
|  |  |  |  | 3 | Используются осмысленные вводные и пояснительные комментарии |  |  |
|  |  | J | Файлы веб-проекта хорошо организованы. |  |  | 2 | 0.50 |
|  |  |  |  | 0 | Файлы не структурированы |  |  |
|  |  |  |  | 1 | Названия файлов указывают на их содержимое |  |  |
|  |  |  |  | 2 | Файлы сгруппированы в каталоги, имена каталогов указывают на их содержимое |  |  |
|  |  |  |  | 3 | Структура файлов позволяет без проблем найти необходимые данные |  |  |
| A3 | Графический дизайн веб-страниц |  |  |  |  |  |  |
|  |  | O | На сайте используется логотип компании |  |  | 3 | 0.25 |
|  |  | O | Используется хотя бы один из предоставленных шрифтов |  |  | 3 | 0.50 |
|  |  | J | Разработанный логотип подчеркивает тему сайта |  |  | 3 | 0.75 |
|  |  |  |  | 0 | Логотип не разработан либо крайне низкого качества |  |  |
|  |  |  |  | 1 | Логотип не подходит под тему сайта |  |  |
|  |  |  |  | 2 | Логотип в целом подходит под тему сайта |  |  |
|  |  |  |  | 3 | Идеальное попадание в тему сайта |  |  |
|  |  | J | Качество реализации логотипа |  |  | 3 | 0.75 |
|  |  |  |  | 0 | Низкий уровень реализации или отсутствие логотипа |  |  |
|  |  |  |  | 1 | Некоторые элементы реализованы хорошо, используются некоторые цвета сайта |  |  |
|  |  |  |  | 2 | Все элементы реализованы хорошо, используются основные цвета сайта |  |  |
|  |  |  |  | 3 | Профессиональная реализация работы |  |  |
|  |  | J | Интерфейс понятен и удобен для использования |  |  | 3 | 1.00 |
|  |  |  |  | 0 | Сайтом крайне сложно пользоваться или работа не представлена |  |  |
|  |  |  |  | 1 | Неудобный интерфейс |  |  |
|  |  |  |  | 2 | Удобный интерфейс |  |  |
|  |  |  |  | 3 | Крайне хорошо проработанный интерфейс, использовать удобно |  |  |
|  |  | J | Учитываются тенденции и требования отрасли к мобильной (смартфон или планшет по заданию) версии страницы |  |  | 3 | 1.00 |
|  |  |  |  | 0 | Мобильная версия не представлена или не учитываются особенности мобильных платформ |  |  |
|  |  |  |  | 1 | Слабо учитываются особенности мобильных платформ |  |  |
|  |  |  |  | 2 | Особенности мобильных платформ отражены, но возможны улучшения |  |  |
|  |  |  |  | 3 | Полностью учитываются особенности мобильных платформ (увеличенный шрифт, измененная структура, крупные элементы для touch-взаимодействия, скрыты некоторые медиа-объекты и т.д.) |  |  |
|  |  | J | Качество реализации страницы "Каталог" |  |  | 3 | 1.50 |
|  |  |  |  | 0 | Страница не создана |  |  |
|  |  |  |  | 1 | Дизайн страницы простой |  |  |
|  |  |  |  | 2 | Дизайн страницы реализован, но стиль отличается от других страниц |  |  |
|  |  |  |  | 3 | Дизайн страницы полностью советует заданию |  |  |
|  |  | J | Визуальная согласованность элементов управления |  |  | 3 | 0.25 |
|  |  |  |  | 0 | Большинство элементов необоснованно отличаются по размеру, оформлению |  |  |
|  |  |  |  | 1 | Некоторые элементы согласованы между собой |  |  |
|  |  |  |  | 2 | Большинство элементов согласованы между собой, но возможны улучшения |  |  |
|  |  |  |  | 3 | Полная согласованность |  |  |
|  |  | J | Идентичность используемых изображений |  |  | 3 | 0.50 |
|  |  |  |  | 0 | Изображения разные, какие-то обработаны, какие-то нет |  |  |
|  |  |  |  | 1 | Некоторые изображения имеют одинаковый стиль |  |  |
|  |  |  |  | 2 | Почти все изображения обработаны в одном стиле |  |  |
|  |  |  |  | 3 | Все изображения отлично обработаны в одинаковом стиле |  |  |
|  |  | J | Единая типографика для всех экранов |  |  | 3 | 0.75 |
|  |  |  |  | 0 | Нормы типографики не соблюдаются |  |  |
|  |  |  |  | 1 | Минимальное соблюдение норм типографики |  |  |
|  |  |  |  | 2 | Нормы типографики соблюдаются, но возможно улучшение |  |  |
|  |  |  |  | 3 | Полное соблюдение норм типографики |  |  |
|  |  | J | Качество информативных сообщений о выполнении каких-либо действий |  |  | 3 | 0.50 |
|  |  |  |  | 0 | Нет или alert |  |  |
|  |  |  |  | 1 | Простые сообщения |  |  |
|  |  |  |  | 2 | Сообщения красивые и вписываются в дизайн |  |  |
|  |  |  |  | 3 | Необычные сообщения, улучшающие впечатление |  |  |
|  |  | J | Визуальная согласованность страниц |  |  | 3 | 0.75 |
|  |  |  |  | 0 | Страницы существенно отличаются по оформлению, размерам элементов и т.д. |  |  |
|  |  |  |  | 1 | Некоторые страницы согласованы между собой |  |  |
|  |  |  |  | 2 | Большинство страниц согласованы между собой, но возможны улучшения |  |  |
|  |  |  |  | 3 | Полная согласованность страниц |  |  |
|  |  | J | Качество реализации админ панели |  |  | 3 | 1.50 |
|  |  |  |  | 0 | Админ панель не создана |  |  |
|  |  |  |  | 1 | Дизайн простой |  |  |
|  |  |  |  | 2 | Дизайн реализован, но стиль отличается от других страниц |  |  |
|  |  |  |  | 3 | Дизайн админ панели полностью соответствует заданию |  |  |
| A4 | Верстка страниц |  |  |  |  |  |  |
|  |  | O | Валидность HTML кода |  | Штраф – 0.25 за каждый тип ошибки | 4 | 0.75 |
|  |  | O | Валидность CSS кода |  | Штраф – 0.5 за каждый тип ошибки | 4 | 1.50 |
|  |  | O | Свёрстан слайдер и девиз компании на главной странице |  | Штраф – 0.4 за отсутствующий элемент | 4 | 0.80 |
|  |  | O | В мобильной версии слайдер отображается корректно |  |  | 4 | 0.50 |
|  |  | O | Свёрстан экран с карточками товара со всеми данными |  | Штраф – 0.3 за отсутствующий элемент | 4 | 0.70 |
|  |  | O | Свёрстан экран с товаром |  | Штраф – 0.25 за каждый пропущенный элемент | 4 | 0.75 |
|  |  | O | Сверстана страница "Где нас найти?" |  | Штраф – 0.3 за отсутствующий элемент | 4 | 0.60 |
|  |  | O | Сверстана кнопка (ссылка) Выход |  |  | 4 | 0.20 |
|  |  | O | Сверстана форма регистрации |  | Штраф за каждый отсутствующий элемент 0.2 | 4 | 0.80 |
|  |  | O | В форме регистрации поля ввода e-mail имеет корректный тип |  | Или проводится валидация адреса e-mail средствами js | 4 | 0.20 |
|  |  | O | Сверстана форма авторизации |  |  | 4 | 0.50 |
|  |  | O | Сверстана админ панель |  | Штраф за каждый отсутствующий элемент 0.25 (просмотр и изменение статуса заказов, удаление, добавление категории) | 4 | 0.75 |
|  |  | O | Сверстана корзина |  | Штраф за каждый отсутствующий элемент 0.3 | 4 | 0.60 |
|  |  | O | Сверстан экран подтверждения заказа |  |  | 4 | 0.25 |
|  |  | O | Сверстана страница просмотра своих заявок |  | Штраф за каждый отсутствующий элемент 0.5 | 4 | 0.25 |
|  |  | O | Сверстаны элементы для фильтрации товаров по категориям |  |  | 4 | 0.25 |
|  |  | O | Сверстаны элементы для упорядочивания товаров |  | Штраф – 0.3 за отсутствующий элемент | 4 | 0.60 |
| A5 | Программирование на стороне клиента |  |  |  |  |  |  |
|  |  | O | Реализованы анимированные сообщения об ошибках при валидации вводимых значений |  | штраф 0,3 без анимации | 5 | 0.50 |
|  |  | O | При ошибке заполнения поле визуально выделяется |  | штраф 0,3, если выделение поля происходит только стандартными возможностями браузера | 5 | 0.50 |
|  |  | O | Проверка валидации полей ФИО выполняется |  |  | 5 | 0.50 |
|  |  | O | Форма регистрации не отправляется, если данные не заполнены |  |  | 5 | 0.25 |
|  |  | O | Проверка на уникальность логина проходит без перезагрузки страницы |  |  | 5 | 0.75 |
|  |  | O | Проверка повтора пароля |  |  | 5 | 0.25 |
|  |  | O | Проверка согласия с правилами регистрации |  |  | 5 | 0.25 |
|  |  | O | Реализована анимация слайдера |  |  | 5 | 0.50 |
|  |  | O | Требования к полям формы регистрации проходят без перезагрузки страницы |  |  | 5 | 0.50 |
|  |  | O | Проверка при добавлении товара в корзину учитывает наличие товаров |  |  | 5 | 0.75 |
|  |  | O | Появляется сообщение об ошибке добавления товара в корзину/корзине |  |  | 5 | 0.50 |
|  |  | O | Подтверждение заказа происходит без перезагрузки страницы |  |  | 5 | 0.20 |
|  |  | O | В слайдере выводиться пять последних товаров |  |  | 5 | 0.50 |
|  |  | O | JS код работает без отображения ошибок |  |  | 5 | 0.30 |
|  |  | O | Реализована сортировка товаров в каталоге |  |  | 5 | 0.75 |
|  |  | O | Кнопка "В корзину" есть только у авторизированного пользователя |  |  | 5 | 0.75 |
|  |  | J | Качество JS кода |  |  | 5 | 1.00 |
|  |  |  |  | 0 | В коде нельзя быстро разобраться |  |  |
|  |  |  |  | 1 | Осмысленное именование переменных, функций |  |  |
|  |  |  |  | 2 | Уровень 1 + Используется js framework или чистый JS (с использованием классов, ООП) |  |  |
|  |  |  |  | 3 | Используется js framework, код организован и легко читаем |  |  |
|  |  | J | Общее впечатление от реализации анимации слайдера |  |  | 5 | 1.25 |
|  |  |  |  | 0 | Отсутствует |  |  |
|  |  |  |  | 1 | Анимация реализована стандартно |  |  |
|  |  |  |  | 2 | Анимация имеет интересные фичи |  |  |
|  |  |  |  | 3 | Уникальная анимация |  |  |
| A6 | Программирование на стороне сервера |  |  |  |  |  |  |
|  |  | O | Данные сохраняются в БД |  |  | 6 | 0.50 |
|  |  | O | Реализована авторизация пользователей |  |  | 6 | 0.50 |
|  |  | O | Админ панель доступна только администратору |  |  | 6 | 0.25 |
|  |  | O | Просмотр своих заявок доступен только зарегистрированному пользователю |  |  | 6 | 0.25 |
|  |  | O | В каталоге не отображаются товары, которых нет в наличии |  |  | 6 | 0.25 |
|  |  | O | При не правильной паре логин/пароль выводится сообщение |  |  | 6 | 0.25 |
|  |  | O | Реализована регистрация |  |  | 6 | 0.75 |
|  |  | O | Проверка валидности логина |  |  | 6 | 0.25 |
|  |  | O | Проверка уникальности логина |  |  | 6 | 0.50 |
|  |  | O | Реализован выход |  |  | 6 | 0.25 |
|  |  | O | Клиент может сделать заказ |  |  | 6 | 0.50 |
|  |  | O | При добавлении товара в корзину учитывается его количество |  |  | 6 | 0.25 |
|  |  | O | При создании заказа в БД создается временная метка |  |  | 6 | 0.25 |
|  |  | O | При создании заказа ему присваивается статус "Новый" |  |  | 6 | 0.25 |
|  |  | O | Пользователь может просмотреть свои заказы |  |  | 6 | 0.50 |
|  |  | O | При отображении заказов сортируются в соответствии с заданием |  |  | 6 | 0.25 |
|  |  | O | Для администратора реализована фильтрация заказов по статусу |  |  | 6 | 0.25 |
|  |  | O | Пользователь может удалить заказы со статусом "Новый" |  |  | 6 | 0.50 |
|  |  | O | Администратор может менять статус заказа на "Отменен" |  | Если комментарий не добавляется в БД штраф 0,25 | 6 | 0.50 |
|  |  | O | Администратор может менять статус заказа на "Подтверждён" |  |  | 6 | 0.50 |
|  |  | O | Администратор может добавить, редактировать и удалять товар |  | Штраф 0,25 за отсутствие функции | 6 | 0.50 |
|  |  | O | Администратор может добавлять новые категории |  |  | 6 | 0.25 |
|  |  | O | Администратор может удалять категории |  |  | 6 | 0.25 |
|  |  | O | При удалении категории администратором удаляются все товары данной категории |  |  | 6 | 0.50 |
|  |  | O | Отсутствуют ошибки, связанные с безопасностью |  | mysql инъекции, XSS-атаки | 6 | 1.00 |
| **Итого** | | | | | | | 44 |

# ПРИМЕР РЕШЕНИЯ ЗАДАНИЯ ДЭМОНСТРАЦИОННОГО ЭКЗАМЕНА

## Установка фреймворка Yii 2

Для локальной установки Yii2 выполните следующие действия:

1. Установите и запустите локальный веб-сервер OSPanel.

OSPanel (Open Server) – это пакет серверного программного обеспечения для организации собственного портативного веб сервера для разработки сайтов.

1. Запустите консоль веб-сервера (Выполните [Меню → Дополнительно → Консоль]).
2. Обновите менеджер пакетов Composer – composer self-update.

Composer — это пакетный менеджер уровня приложений для языка программирования PHP, который предоставляет средства по управлению зависимостями в PHP-приложении. Composer работает через интерфейс командной строки и устанавливает зависимости (например библиотеки) для приложения. Он также позволяет пользователям устанавливать PHP-приложения, которые доступны на packagist.org, который является его основным репозиторием, где содержатся все доступные пакеты.

1. Перейдите в каталог domains – cd domains.
2. Запустите установку фреймворка Yii – composer create-project yiisoft/yii2-app-basic xxxxx-m4.wsr.ru, где xxxxx – ваш логин, перезагрузите OSPanel.

Для удаленной установки Yii2 на сервере колледжа выполните следующие действия:

1. Зайдите в личный кабинет по адресу <http://lk.wsr.ru/> используя данные с карточек.
2. Подключите сетевой диск Y, для этого запустите командную строку CMD и выполните представленную в личном кабинете команду net use.
3. Скачайте из личного кабинета архив с Yii2.
4. Перейдите в каталог Y:\xxxxx-m4.wsr.ru\, где xxxxx – ваш логин.
5. Распакуйте содержимое архива (все файлы и папки внутри basic) в текущий каталог.

## Первоначальная настройка фреймворка Yii 2

Для первоначальной настройки фреймворка Yii2 выполните следующие действия:

1. В корневой директории вашего проекта создайте файл .htaccess со следующим содержимым:

RewriteEngine on

RewriteRule (.\*)? /web/$1

Данная настройка перенаправит все запросы, идущие на фреймворк, в папку web – точку входа в приложение.

1. Откройте файл assets\AppAsset.php – основной пакет активов приложения и подключите к своему проекту свой файл каскадных таблиц стилей и свой файл с javascript кодом (не забудьте создать данные файлы и папки в директории web).

class AppAsset extends AssetBundle

{

public $basePath = '@webroot';

public $baseUrl = '@web';

public $css = [

'css/site.css',

*/\* Подключаем свой CSS \*/*

'css/my.css',

];

public $js = [

*/\* Подключаем свой JS \*/*

'js/my.js',

];

public $depends = [

'yii\web\YiiAsset',

'yii\bootstrap5\BootstrapAsset'

];

}

1. Откройте файл конфигурации подключения к базе данных config\db.php и укажите настройки для подключения к базе данных.

<?php

return [

'class' => 'yii\db\Connection',

'dsn' => 'mysql:host=localhost;dbname=ИмяБазыДанных',

'username' => 'ЛогинПользователяБД',

'password' => 'ПарольПользователяБД',

'charset' => 'utf8',

];

1. Откройте файл конфигурации веб-приложения config/web.php и установите значение текущего языка локализации сообщений на русский язык, задайте случайный набор символов параметру валидации cookieValidationKey – это дополнительная безопасность проверки cookie, которая защищает файлы cookie от изменения на стороне клиента, объявите параметр базового URL-адрес проекта baseUrl. Далее раскомментируйте параметр urlManager, который позволяет собирать URL динамически на основе правил маршрутизации и параметр allowedIPs раздела конфигурации gii модуля, отвечающий за открытые доступа к gii с конкретных IP адресов.

$config = [

'id' => 'basic',

'basePath' => *dirname*(*\_\_DIR\_\_*),

'bootstrap' => ['log'],

'aliases' => [

'@bower' => '@vendor/bower-asset',

'@npm' => '@vendor/npm-asset',

],

*/\*Добовляем русскую локализацию \*/*

'language' => 'ru-Ru',

'components' => [

'request' => [

*/\* Вставляем секретный ключ валидации cookie \*/*

'cookieValidationKey' => 'randomString',

*/\* Указываем базовый адрес проекта \*/*

'baseUrl' => '',

],

'cache' => [

'class' => 'yii\caching\FileCache',

],

'user' => [

'identityClass' => 'app\models\User',

'enableAutoLogin' => true,

],

'errorHandler' => [

'errorAction' => 'site/error',

],

'mailer' => [

'class' => \yii\symfonymailer\Mailer::class,

'viewPath' => '@app/mail',

*// send all mails to a file by default.*

'useFileTransport' => true,

],

'log' => [

'traceLevel' => *YII\_DEBUG* ? 3 : 0,

'targets' => [

[

'class' => 'yii\log\FileTarget',

'levels' => ['error', 'warning'],

],

],

],

'db' => $db,

*/\* Включаем ЧПУ \*/*

'urlManager' => [

'enablePrettyUrl' => true,

'showScriptName' => false,

'rules' => [

],

],

*/\* \*/*

],

'params' => $params,

];

if (*YII\_ENV\_DEV*) {

*// configuration adjustments for 'dev' environment*

$config['bootstrap'][] = 'debug';

$config['modules']['debug'] = [

'class' => 'yii\debug\Module',

*// uncomment the following to add your IP if you are not connecting from localhost.*

*//'allowedIPs' => ['127.0.0.1', '::1'],*

];

$config['bootstrap'][] = 'gii';

$config['modules']['gii'] = [

'class' => 'yii\gii\Module',

*// Добавляем IP адрес ПК для доступа к генератору Gii*

'allowedIPs' => ['\*', '::1'],

];

}

1. Скопируйте выбранный вами файл шрифта из представленных медиафайлов в папку со шрифтами в директорию web\css\fonts\ .
2. Разработайте логотип компании размером 200x50 пикселей c прозрачным фоном и экспортируйте его в формате png в директорию web\images\logo.png .
3. Создайте папку web\uploads – в эту папку будут загружаться изображения товаров.
4. Переименуйте стандартную модель models\User.php в models\UserOld.php , не забудьте также переименовать название класса с User на UserOld.

**При выполнении операции переименования через PhpStorm, используя команды Refactor → Rename, обязательно снимите все чек-боксы.**

1. Откройте ваш файл с каскадными таблицами стилей web\css\my.css и напишите следующие правила оформления

*/\* Подключаем внешний файл шрифта \*/*

@font-face {

font-family: "Ek Mukta";

src: url("fonts/EkMuktaRegular.ttf");

}

*/\* Используем внешний шрифт для всех элементов \*/*

\* {

font-family: "Ek Mukta";

}

*/\* Выделяем поля не прошедшие валидацию данных \*/*

input[aria-invalid="true"] {

border-color: red;

}

*/\* Меняем цвет у ошибок валидации в форме\*/*

.help-block {

color: red;

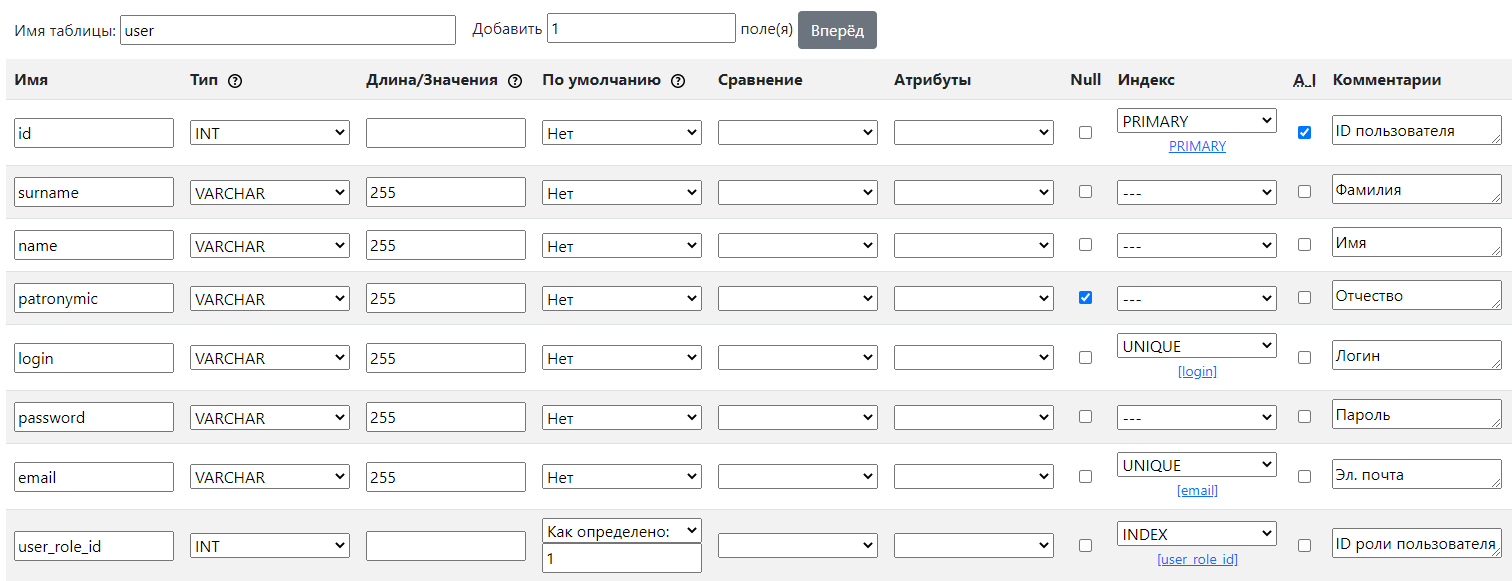
}

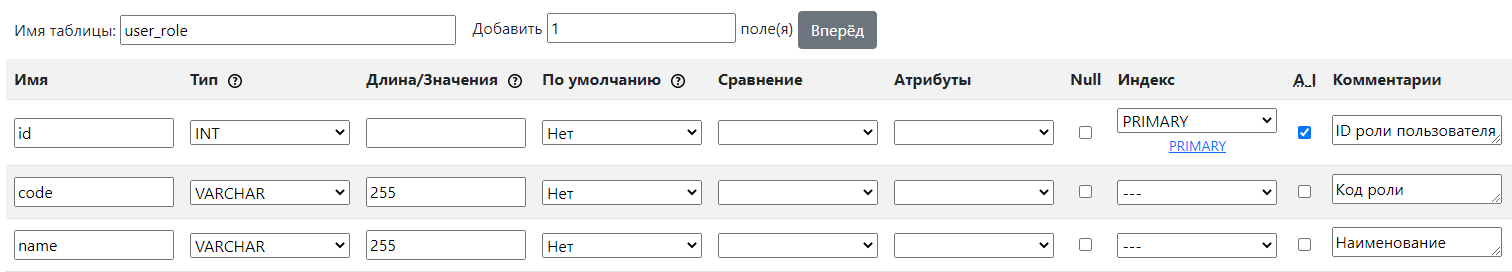
## Разработка базы данных

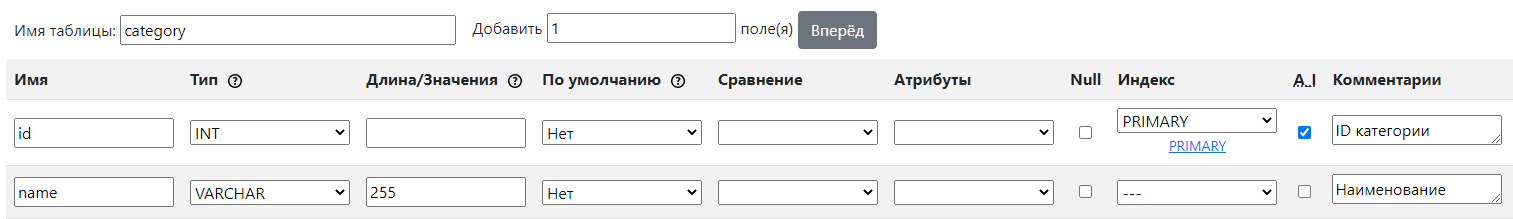
Изучив задание демонстрационного экзамена, можно выделить 8 сущностей:

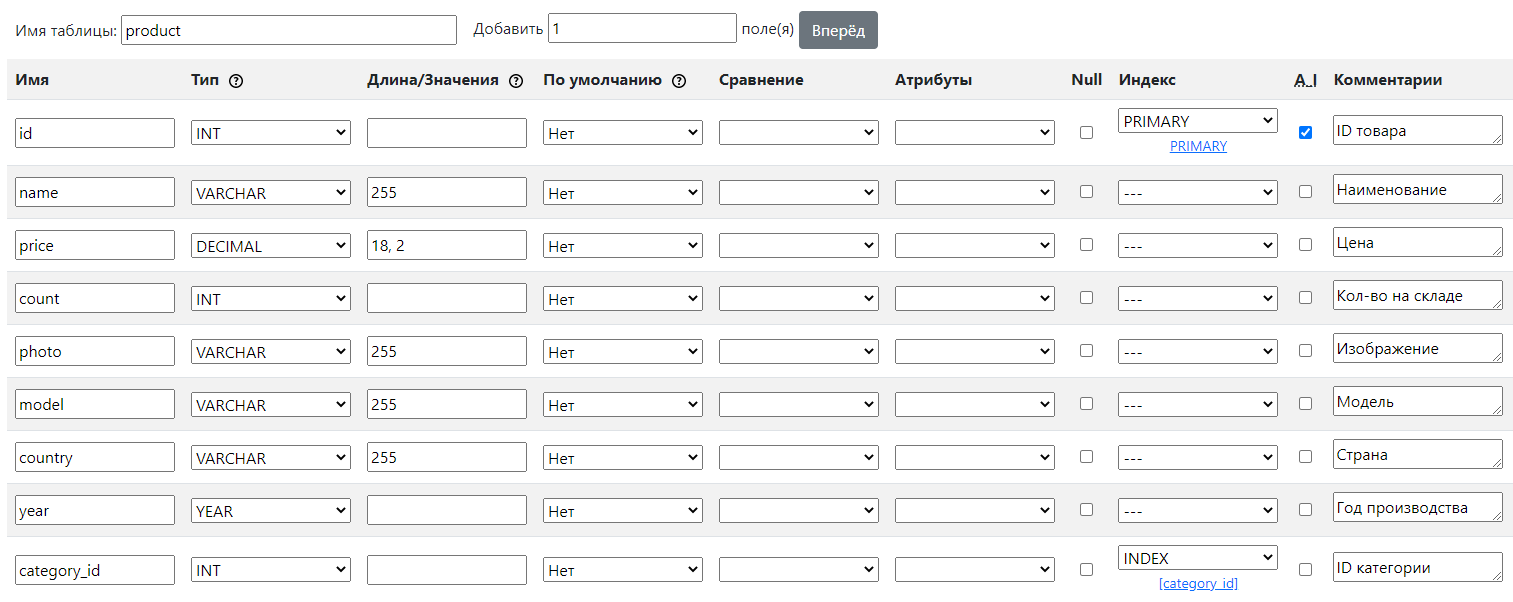
1. Сущность «Пользователь» с атрибутами «фамилия», «имя», «отчество», «логин», «пароль», «эл. почта».
2. Сущность «Роль пользователя» с атрибутами «код роли» и «наименование».
3. Сущность «Категория товара» с атрибутом «название».
4. Сущность «Товар» с атрибутами «наименование», «цена», «количество», «изображение», «модель», «страна производитель», «год производства».
5. Сущность «Корзина» с атрибутом «количество».
6. Сущность «Заказ» с атрибутами «дата/время заказа», «причина отказа».
7. Сущность «Состав заказа» с атрибутами «количество» и «цена».
8. Сущность «Статус заказа» с атрибутами «код статуса» и «наименование статуса заказа»

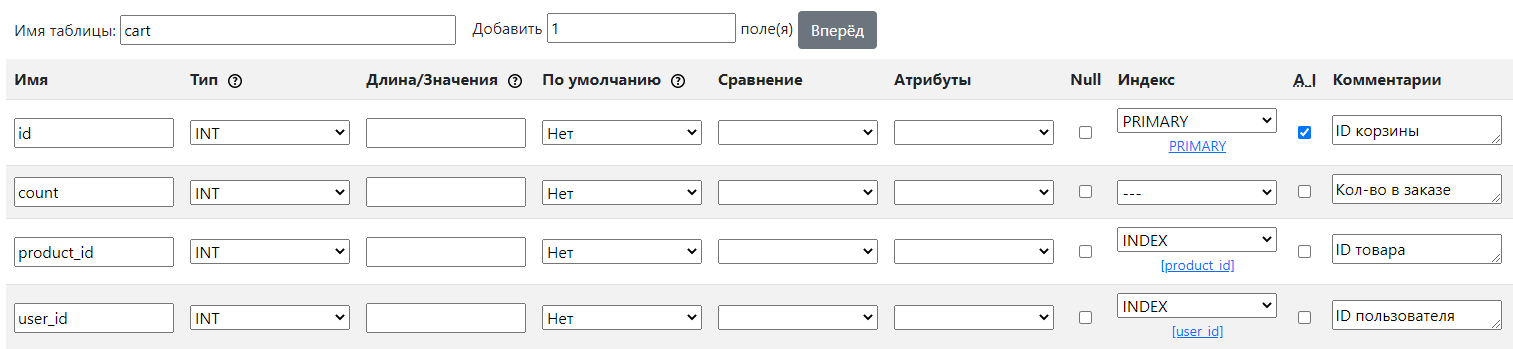
Откройте СУБД phpMyAdmin и создайте 8 таблиц:

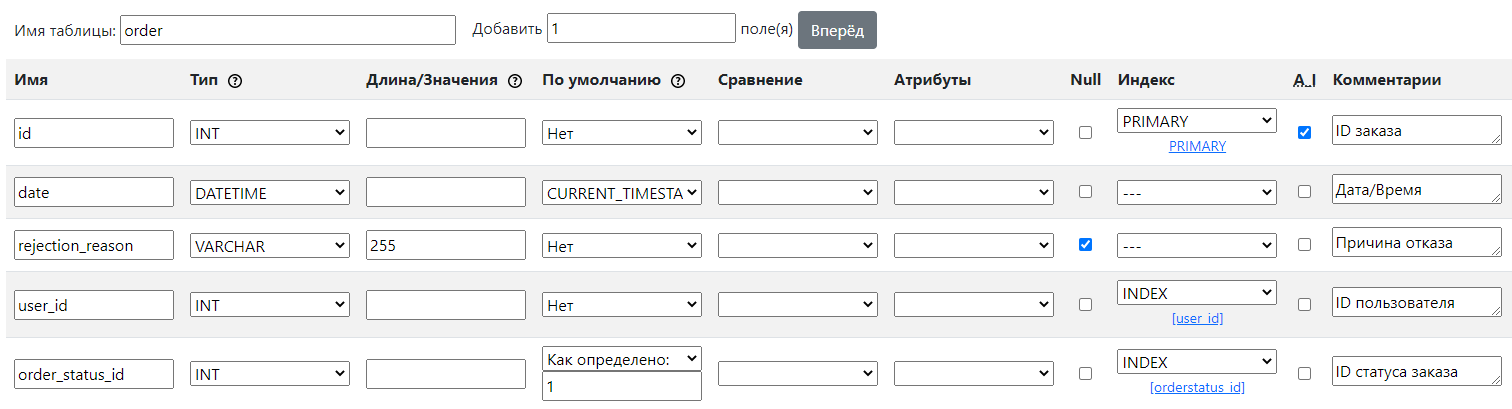
****

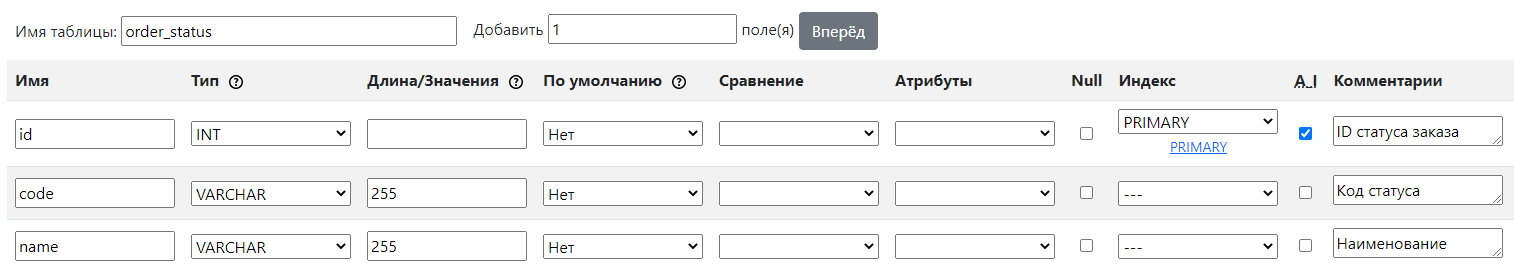
****

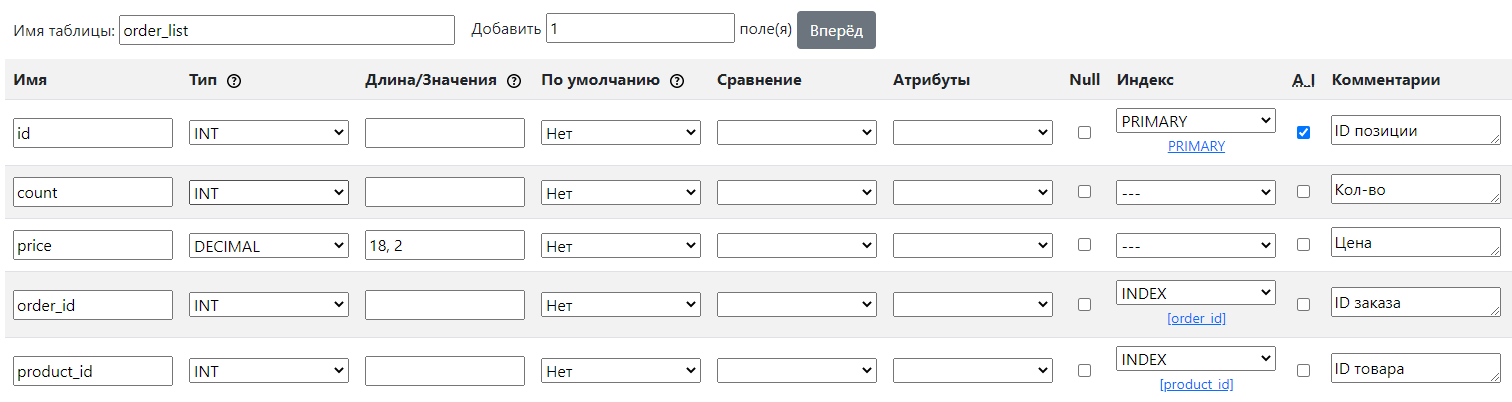
****



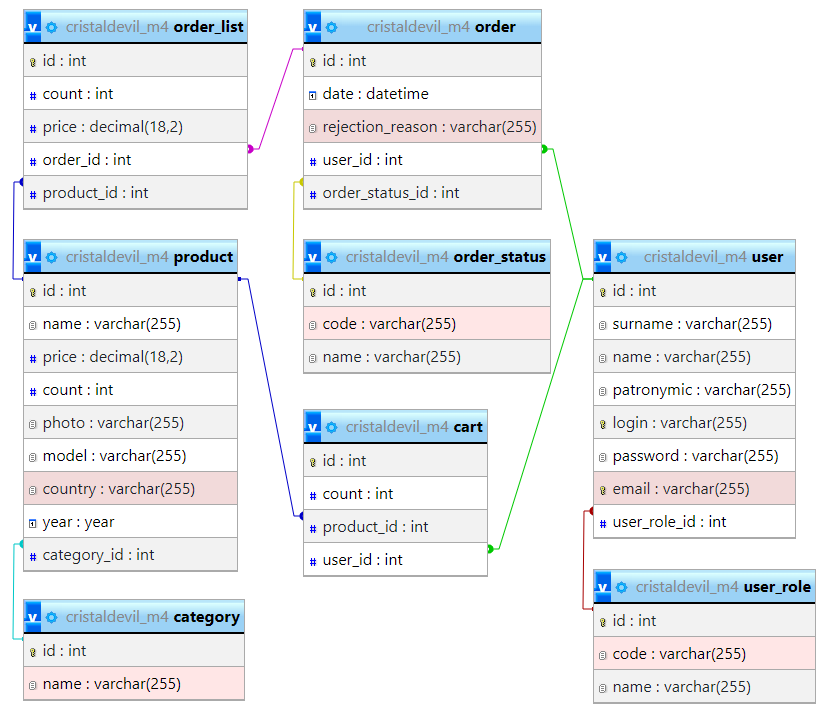








В режиме дизайнера установите связи между таблицами, тип связи – CASCADE.



В таблицу «user\_role» добавьте две записи.

|  |  |  |
| --- | --- | --- |
| **id** | **code** | **name** |
| 1 | user | Пользователь |
| 2 | admin | Администратор |

В таблицу «user» добавьте администратора с логином и паролем из задания (логин – admin, пароль – admin11). Не забудьте указать для поля user\_role\_id значение 2.

В таблицу «category» добавьте три записи.

|  |  |
| --- | --- |
| **id** | **name** |
| 1 | Лазерные принтеры |
| 2 | Струйные принтеры |
| 3 | Термопринтеры |

В таблицу «order\_status» добавьте три записи.

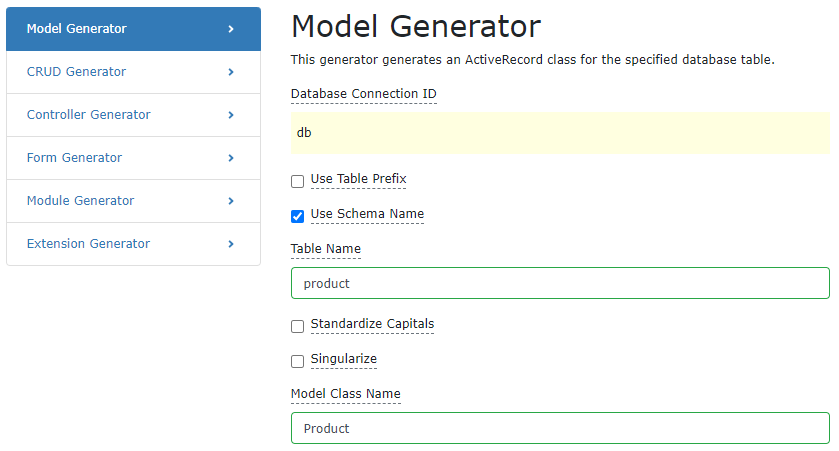
|  |  |  |
| --- | --- | --- |
| **id** | **code** | **name** |
| 1 | new | Новый |
| 2 | confirm | Подтвержденный |
| 3 | reject | Отклоненный |

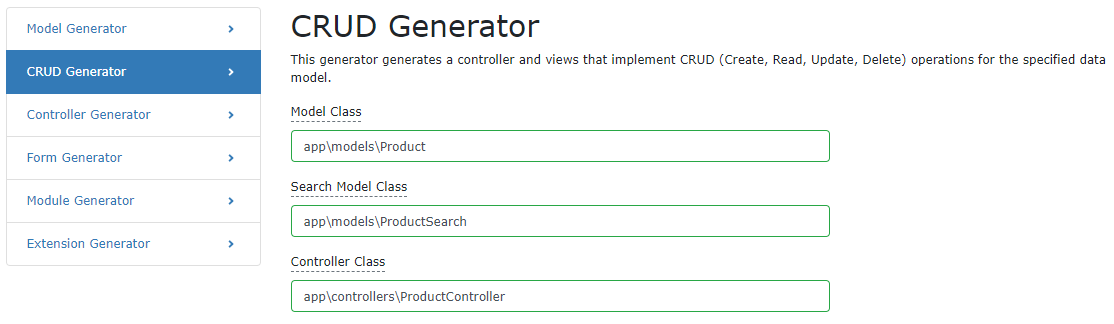
## GII

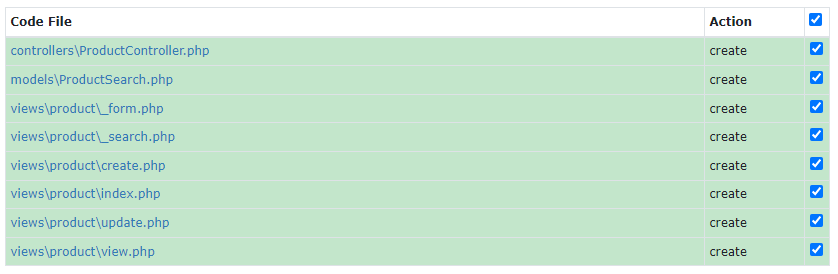
Зайдите в Gii по адресу: http://xxxxx-m4.wsr.ru/gii, где xxxxx – ваш логин.

Перейдите в Model Generator и сгенерируйте модели для всех таблиц. Перейдите в CRUD Generator и сгенерируйте контроллеры и представления для следующих моделей данных (для Order и Product необходимо указать Search Model Class):

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Имя модели** | **Имя модели поиска** | **Имя контроллера** | **\_form** | **\_search** | **create** | **index** | **update** | **view** |
| Cart | – | CartController | + | – | – | + | – | – |
| Category | – | CategoryController | + | – | + | + | + | + |
| Order | OrderSearch | OrderController | + | + | – | + | + | – |
| OrderList | – | – | – | – | – | – | – | – |
| OrderStatus | – | – | – | – | – | – | – | – |
| Product | ProductSearch | ProductController | + | + | + | + | + | + |
| User | – | UserController | + | – | + | + | + | + |
| UserRole | – | – | – | – | – | – | – | – |







## Авторизация

В модели models\User.php укажите что класс User реализуется с помощью интерфейса IdentityInterface, напишите реализацию наследуемых методов данного интерфейса (все методы копируем с модели UserOld). Напишите метод isAdmin для определения роли администратора.

class User extends \yii\db\ActiveRecord implements \yii\web\IdentityInterface

{

*/\* Реализуем методы наследуемого интерфейса \*/*

public function getId()

{

return $this->id;

}

public function validateAuthKey($authKey)

{

return false;

}

public function getAuthKey()

{

return null;

}

public static function findIdentity($id)

{

return static::*findOne*($id);

}

public static function findIdentityByAccessToken($token, $type = null)

{

return null;

}

*/\* Поиск пользователя по логину и проверка введенного пароля \*/*

public static function findByUsername($login)

{

return User::*findOne*(['login' => $login]);

}

public function validatePassword($password)

{

return $this->password === $password;

}

*/\* Проверка на вход администратора \*/*

public function isAdmin(){

return $this->userRole->code === "admin";

}

В файле шаблона views\layouts\main.php на 50 строке найдите строку Yii::$app->user->identity->username и измените на Yii::$app->user->identity->email (либо на любой другой существующий атрибут), чтобы избавиться от ошибки, так как атрибута username нет в модели User.

Русифицируйте поля меток в форме авторизации, для этого допишите метод attributeLabels() в модели LoginForm.

class LoginForm extends Model

{

*/\* Описание названий меток \*/*

public function attributeLabels()

{

return [

'username' => 'Логин',

'password' => 'Пароль',

'rememberMe' => 'Запомнить меня',

];

}

При авторизации администратора перенаправьте его в админ панель, смените редиректор в контроллере controllers\SiteController.php в методе actionLogon.

public function actionLogin()

{

if (!Yii::*$app*->user->isGuest) {

return $this->goHome();

}

$model = new LoginForm();

if ($model->load(Yii::*$app*->request->post()) && $model->login()) {

*/\* Если администратор, перенаправляем в админ-панель \*/*

if (Yii::*$app*->user->identity->isAdmin()){

return $this->redirect(['/admin']);

}

return $this->goBack();

}

$model->password = '';

return $this->render('login', [

'model' => $model,

]);

}

Русифицируйте и отредактируйте представление формы входа views\site\login.php .

## Регистрация

В модели models\User.php добавьте переменные для полей подтверждения пароля и согласия с правилами регистрации на сайте. Также добавьте подписи к новым полям и русифицируйте используемые подписи в форме.

*/\* Добавляем переменные для полей подтверждения пароля и согласия \*/*

public $password\_repeat;

public $rule;

*/\* Описание названий меток \*/*

public function attributeLabels()

{

return [

'id' => 'ID',

'surname' => 'Фамилия',

'name' => 'Имя',

'patronymic' => 'Отчество',

'login' => 'Логин',

'password' => 'Пароль',

'email' => 'Почта',

'user\_role\_id' => 'User Role ID',

*//Добавляем метки для полей password\_repeat и rule*

'password\_repeat' => 'Повтор пароля',

'rule' => 'Я согласен с правила регистрации на сайте',

];

}

Далее добавьте правила валидации полей из задания.

*/\* Правила валидации данных \*/*

public function rules()

{

return [

[['surname', 'name', 'login', 'password', 'email'], 'required'],

[['user\_role\_id'], 'integer'],

[['surname', 'name', 'patronymic', 'login', 'password', 'email'], 'string', 'max' => 255],

[['login'], 'unique'],

[['email'], 'unique'],

[['user\_role\_id'], 'exist', 'skipOnError' => true, 'targetClass' => UserRole::class, 'targetAttribute' => ['user\_role\_id' => 'id']],

*/\* Дописываем правила валидации по заданию \*/*

[['surname', 'name', 'patronymic'], 'match', 'pattern' => '/^[а-яА-ЯёЁ -]+$/u', 'message' => 'Разрешены русские буквы, пробел и тире'],

[['login'], 'match', 'pattern' => '/^[a-zA-Z0-9-]+$/i', 'message' => 'Разрешены латинские буквы, цифры и тире'],

[['email'], 'email'],

[['password\_repeat'], 'compare', 'compareAttribute' => 'password', 'message' => 'Пароли не совпадают'],

[['rule'], 'compare', 'compareValue' => 1, 'message' => 'Согласитесь с правилами регистрации на сайте'],

];

}

После регистрации перенаправьте пользователя на страницу авторизации, для этого смените редиректор в контроллере controllers\UserController.php в методе actionCreate.

public function actionCreate()

{

$model = new User();

if ($this->request->isPost) {

if ($model->load($this->request->post()) && $model->save()) {

return $this->redirect(['/site/login']);

}

} else {

$model->loadDefaultValues();

}

return $this->render('create', [

'model' => $model,

]);

}

В файле представления views\user\\_form.php отредактируйте форму, удалите поле id\_role и добавьте два поля для подтверждения пароля и согласия с правила регистрации на сайте. Добавьте возможность проверки на уникальность логина и пароля.

<?php $form = ActiveForm::*begin*(); ?>

<?= $form->field($model, 'surname')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'name')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'patronymic')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'login', ['enableAjaxValidation' => true])->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'password')->passwordInput(['maxlength' => true]) ?>

<?= $form->field($model, 'password\_repeat')->passwordInput(['maxlength' => true]) ?>

<?= $form->field($model, 'email', ['enableAjaxValidation' => true])->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'rule')->checkbox() ?>

<div class="form-group">

<?= Html::*submitButton*('Сохранить', ['class' => 'btn btn-success']) ?>

</div>

<?php ActiveForm::*end*(); ?>

В контроллере controllers\UserController.php в методе actionCreate добавьте условие получение AJAX и POST запроса для выполнения валидации формы.

public function actionCreate()

{

$model = new User();

*/\* Проверка валидации по Ajax \*/*

if (\Yii::*$app*->request->isAjax && $model->load(\Yii::*$app*->request->post())){

\Yii::*$app*->response->format = \yii\web\Response::*FORMAT\_JSON*;

return \yii\bootstrap5\ActiveForm::*validate*($model);

}

Русифицируйте все представления для user.

Разрешите просмотр, удаление и обновление пользователей только администраторам, для этого измените метод behaviors в контроллере controllers\UserController.php (метод можно скопировать из SiteController, изменив действия и дописав параметр matchCallback).

class UserController extends Controller

{

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['index', 'view', 'update', 'delete'],

'rules' => [

[

'actions' => ['index', 'view', 'update', 'delete'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return \Yii::*$app*->user->identity->isAdmin();

}

],

],

],

];

}

## Меню сайта

Откройте файл шаблона views\layouts\main.php , добавьте в меню навигации логотип компании и измените меню сайта согласно заданию, с учетом роли пользователя. Добавьте блок для информации (необходим для оповещений в разделе в каталог, корзина и т.д.).

<?php

NavBar::*begin*([

'brandLabel' => '<img src="http://' . $\_SERVER["SERVER\_NAME"] . '/images/logo.png" alt="Логотип" height="30" />',

'brandUrl' => Yii::*$app*->homeUrl,

'options' => ['class' => 'navbar-expand-md navbar-dark bg-dark fixed-top']

]);

echo Nav::*widget*([

'options' => ['class' => 'navbar-nav'],

'items' => [

*/\* Меню для всех \*/*

['label' => 'О нас', 'url' => ['/site/index']],

['label' => 'Каталог', 'url' => ['/product/index']],

['label' => 'Где нас найти?', 'url' => ['/site/contact']],

*/\* Меню для админа \*/*

['label' => 'Админ панель', 'url' => ['/admin/index'], 'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()],

['label' => 'Пользователи', 'url' => ['/user/index'], 'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()],

['label' => 'Категории', 'url' => ['/category/index'], 'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()],

['label' => 'Товары', 'url' => ['/product/index'], 'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()],

['label' => 'Заказы', 'url' => ['/order/index'], 'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()],

*/\* Меню для пользователя \*/*

['label' => 'Корзина', 'url' => ['/cart/index'], 'visible' => !Yii::*$app*->user->isGuest && !Yii::*$app*->user->identity->isAdmin()],

['label' => 'Мои заказы', 'url' => ['/order/index'], 'visible' => !Yii::*$app*->user->isGuest && !Yii::*$app*->user->identity->isAdmin()],

*/\* Регистрация и вход \*/*

['label' => 'Регистрация', 'url' => ['/user/create'], 'visible' => Yii::*$app*->user->isGuest],

Yii::*$app*->user->isGuest

? ['label' => 'Вход', 'url' => ['/site/login']]

: '<li class="nav-item">'

. Html::*beginForm*(['/site/logout'])

. Html::*submitButton*(

'Выход (' . Yii::*$app*->user->identity->email . ')',

['class' => 'nav-link btn btn-link logout']

)

. Html::*endForm*()

. '</li>'

]

]);

?>

<div id="info-message" style="color: white"></div>

<?php

NavBar::*end*();

?>

## Админ панель

Создайте контроллер controllers\AdminController.php .

<?php

namespace app\controllers;

use yii\web\Controller;

class AdminController extends Controller

{

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['index'],

'rules' => [

[

'actions' => ['index'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return \Yii::*$app*->user->identity->isAdmin();

}

],

],

],

];

}

*/\* Маршрут admin/index отрисовывает представление view/admin/index.php \*/*

public function actionIndex(){

return $this->render('index');

}

}

Создайте представление view\admin\index.php .

<?php

use yii\helpers\Html;

?>

<section>

<h2>Админ панель</h2>

<div>

<?= Html::*a*('Работа с пользователя', ['/user'], ['class' => 'btn btn-success']) ?>

<?= Html::*a*('Работа с категориями', ['/category'], ['class' => 'btn btn-success']) ?>

<?= Html::*a*('Работа с товарами', ['/product'], ['class' => 'btn btn-success']) ?>

<?= Html::*a*('Работа с заказами', ['/order'], ['class' => 'btn btn-success']) ?>

</div>

</section>

## Категории товаров

Русифицируйте представления и метки в модели (метод attributeLabels()).

Разрешите создание, просмотр, удаление и обновление категорий только администраторам, для этого в контроллере controllers\CategoryController.php измените метод behaviors().

class CategoryController extends Controller

{

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['create', 'index', 'view', 'update', 'delete'],

'rules' => [

[

'actions' => ['create', 'index', 'view', 'update', 'delete'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return \Yii::*$app*->user->identity->isAdmin();

}

],

],

],

];

}

## Товары

Русифицируйте представления и метки в модели (метод attributeLabels()).

В модели models\Product.php добавьте правило валидации для загрузки изображений.

*/\* Правила валидации данных \*/*

public function rules()

{

return [

[['name', 'price', 'count', 'photo', 'model', 'country', 'year', 'id\_category'], 'required'],

[['price'], 'number'],

[['count', 'id\_category'], 'integer'],

[['year'], 'safe'],

[['name', 'model', 'country'], 'string', 'max' => 255],

[['id\_category'], 'exist', 'skipOnError' => true, 'targetClass' => Category::class, 'targetAttribute' => ['id\_category' => 'id']],

*//Добавляем правило валидации для загрузки изображений*

[['photo'], 'file', 'skipOnEmpty' => false, 'extensions' => 'png, jpg', 'message' => 'Разрешены форматы PNG, JPG'],

];

}

Далее в модели добавьте метод для загрузки изображений.

*/\* Метод загрузки изображений в директорию uploads \*/*

public function upload()

{

if ($this->validate()) {

$fileName = 'uploads/' . $this->photo->baseName . '.' . $this->photo->extension;

$this->photo->saveAs($fileName);

return true;

} else {

return false;

}

}

Создайте папку uploads в директории web.

Разрешите создание (create), обновление (update) и удаление (delete) товаров только администраторам, для этого в контроллере controllers\ProductController.php измените метод behaviors().

class ProductController extends Controller

{

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['create', 'update', 'delete'],

'rules' => [

[

'actions' => ['create', 'update', 'delete'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return \Yii::*$app*->user->identity->isAdmin();

}

],

],

],

];

}

Далее в контроллере измените метод actionCreate(), который перед созданием товара будет загружать изображение на сервер.

public function actionCreate()

{

$model = new Product();

if ($this->request->isPost) {

if ($model->load($this->request->post())) {

*/\* Сохранение фотографии перед добавлением товара \*/*

$model->photo = \yii\web\UploadedFile::*getInstance*($model, 'photo');

if ($model->upload() && $model->save(false)) {

return $this->redirect(['index']);

}

}

} else {

$model->loadDefaultValues();

}

return $this->render('create', [

'model' => $model,

]);

}

Аналогично измените метод actionUpdate(), который позволяет редактировать данные.

public function actionUpdate($id)

{

$model = $this->findModel($id);

if ($this->request->isPost) {

if ($model->load($this->request->post())) {

/\* Сохранение фотографии перед обновлением товара \*/

$model->photo = \yii\web\UploadedFile::*getInstance*($model, 'photo');

if ($model->upload() && $model->save(false)) {

return $this->redirect(['index']);

}

}

}

return $this->render('update', [

'model' => $model,

]);

}

Измените представление view\product\\_form.php , добавив возможность выбора категории из списка, изменив поле для ввода id\_category, а также изменив тип поля для изображения товара.

<?php $form = ActiveForm::*begin*(); ?>

<?= $form->field($model, 'name')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'price')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'count')->textInput() ?>

<?= $form->field($model, 'photo')->fileInput(['maxlength' => true]) ?>

<?= $form->field($model, 'model')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'country')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'year')->textInput() ?>

<?= $form->field($model, 'id\_category')

->dropDownList(\yii\helpers\ArrayHelper::*map*(\app\models\Category::*find*()

->all(), 'id', 'name')) ?>

<div class="form-group">

<?= Html::*submitButton*('Сохранить', ['class' => 'btn btn-success']) ?>

</div>

<?php ActiveForm::*end*(); ?>

Русифицируйте модель и представления product.

public function attributeLabels()

{

return [

'id' => 'ID',

'name' => 'Наименование',

'price' => 'Цена',

'count' => 'Количество на складе',

'photo' => 'Изображение',

'model' => 'Название модели',

'country' => 'Страна производитель',

'year' => 'Год производства',

'category\_id' => 'Категория товара',

];

}

Используйте Админ-панель и добавьте 6 товаров с их описанием.

Измените виджет GridView в представление view\product\index.php .

<?php \yii\widgets\Pjax::*begin*(['id' => 'cart']);?>

<div class="product-index">

<h1><?= Html::*encode*($this->title) ?></h1>

*<!-- Кнопка Добавить товар только для админа-->*

<p>

<?php

if (!Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()){

echo Html::*a*('Добавить товар', ['create'], ['class' => 'btn btn-success']);

} ?>

</p>

<?php *// echo $this->render('\_search', ['model' => $searchModel]);* ?>

<?= GridView::*widget*([

'dataProvider' => $dataProvider,

'filterModel' => $searchModel,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

'name',

'count',

'price',

'country',

'year',

*// Поле для вывода категории*

[

'attribute' => 'category\_id',

'label' => 'Категория',

'format' => 'html',

'value' => function($model){

return $model->category->name;

},

],

*// Поле для вывода фото*

[

'label' => 'Фото товара',

'format' => 'html',

'value' => function($model) {

return Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/uploads/' . $model->photo, ['width' => 200, 'alt' => $model->name, 'title' => $model->name]);

}

],

*// Поле для вывода кнопки Добавить в корзину для пользователя*

[

'label' => 'Добавить в корзину',

'format' => 'raw',

'value' => function($model) {

return Html::*button*('Добавить в корзину', ['class' => 'btn btn-success', 'onclick' => 'addCart('.$model->id.')']);

},

'visible' => !Yii::*$app*->user->isGuest && !Yii::*$app*->user->identity->isAdmin()

],

*// Поле для вывода кнопки Пoдробнее для всех*

[

'label' => 'Подробнее',

'format' => 'raw',

'value' => function($model) {

return Html::*a*('Пoдробнее', ['view', 'id' => $model->id], ['class' => 'btn btn-success']);

}

],

*// Поле для вывода кнопок управления для админа*

[

'class' => ActionColumn::*className*(),

'urlCreator' => function ($action, Product $model, $key, $index, $column) {

return Url::*toRoute*([$action, 'id' => $model->id]);

},

'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()

],

],

]); ?>

</div>

<?php \yii\widgets\Pjax::*end*()?>

Измените виджет GridView в представление view\product\view.php .

<div class="product-view">

<h1><?= Html::*encode*($this->title) ?></h1>

*<!-- Кнопка Добавить товар только для админа-->*

<p>

<?php

if (!Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()) {

echo Html::*a*('Обновить', ['update', 'id' => $model->id], ['class' => 'btn btn-primary']);

echo Html::*a*('Удалить', ['delete', 'id' => $model->id], [

'class' => 'btn btn-danger',

'data' => [

'confirm' => 'Удалить текущий товар?',

'method' => 'post',

],

]);

}

?>

</p>

<?= DetailView::*widget*([

'model' => $model,

'attributes' => [

'id',

'name',

'price',

'count',

*// Поле для вывода фото*

[

'label' => 'Фото товара',

'format' => 'html',

'value' => function($model) {

return Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/uploads/' . $model->photo, ['width' => 200, 'alt' => $model->name, 'title' => $model->name]);

}

],

'model',

'country',

'year',

*// Поле для вывода категории*

[

'attribute' => 'category\_id',

'label' => 'Категория',

'format' => 'html',

'value' => function($model){

return $model->category->name;

},

],

[

'label' => 'Добавить в корзину',

'format' => 'raw',

'value' => function($model) {

return Html::*button*('Добавить в корзину', ['class' => 'btn btn-success', 'onclick' => 'addCart('.$model->id.')']);

},

'visible' => !Yii::*$app*->user->isGuest && !Yii::*$app*->user->identity->isAdmin()

],

],

]) ?>

</div>

В модели models\ProductSearch.php измените запрос на поиск товаров, чтобы он выводил только товары имеющиеся в наличие и сортировал их от новых к старым.

public function search($params)

{

$query = Product::*find*()->where(['>', 'count', 0])->orderBy(['id' => *SORT\_DESC*]);

## Страница «О нас»

В контроллере controllers\SiteController.php измените метод actionIndex().

public function actionIndex()

{

$items = [];

foreach(\app\models\Product::*find*()->orderBy(['id' => *SORT\_DESC*])->limit(5)->all() as $product){

$items[] = [

'content' => \yii\helpers\Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/uploads/' . $product->photo, ['height' => '300', 'style' => 'display: block; margin: 0 auto; border-radius: 20px']),

'caption' => $product->name . '<br>' . $product->price,

'options' => ['style' => 'height:400px;']

];

}

return $this->render('index',['items'=>$items]);

}

Отредактируйте представление view\site\index.php .

<?php

$this->title = 'Интернет магазин Copy Star';

?>

<div class="site-index">

<?= \yii\helpers\Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/images/logo.png', ['width' => 200, 'alt' => 'Логотип']) ?>

<?= \yii\helpers\Html::*tag*('h3', 'Мы можем обслужить Вас быстро, качественно и недорого. Выбирайте любые два пункта!') ?>

<div class="row slider">

<?php echo \yii\bootstrap5\Carousel::*widget*([

'items' => $items,

'options' => [

'style' => 'width:100%; padding-top:50px; background: red;'

]

]);

?>

</div>

</div>

## Страница «Где нас найти?»

Скопируйте карту из медиа архива в папку web\images\map.jpg и отредактируйте представление view\site\contact.php .

<?php

$this->title = 'Где нас найти?';

$this->params['breadcrumbs'][] = $this->title;

?>

<div class="site-contact">

<div class="d-flex">

<div>

<?= \yii\helpers\Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/images/map.jpg', ['width' => 800, 'alt' => 'Карта']) ?>

</div>

<div class="p-4">

<?= \yii\helpers\Html::*tag*('p', 'Адрес: г. Томск, ул. Неизвестная 20') ?>

<?= \yii\helpers\Html::*tag*('p', 'Номер телефона: +7 (913)-114 10-17') ?>

<?= \yii\helpers\Html::*tag*('p', 'Email: copy-star@mail.ru') ?>

</div>

</div>

</div>

## Корзина

Разрешите просмотр корзины только авторизированным пользователям, для этого в контроллере controllers\CartController.php измените метод behaviors().

class CartController extends Controller

{

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['index'],

'rules' => [

[

'actions' => ['index'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return !\Yii::*$app*->user->isGuest && !\Yii::*$app*->user->identity->isAdmin();

}

],

],

],

];

}

В методе actionIndex() измените параметр поиска данных в таблице cart, чтобы выводились данные только текущего пользователя.

public function actionIndex()

{

$dataProvider = new ActiveDataProvider([

'query' => Cart::*find*()->where(['user\_id' => \Yii::*$app*->user->getId()]),

Измените представление view\cart\\_form.php и русифицируйте метки. Виджет Pjax необходим для обновления блоков без перезагрузки страницы, после выполнения ajax запросов (изменение количества заказанного товара, удаление товара из корзины). Добавьте кнопки + и – для добавления и уменьшения (удаления) товара в корзине. Ниже добавьте кнопку оформить заказ и поле для ввода пароля пользователя, который необходимо ввести прежде, чем оформить заказ.

*<!-- Pjax для обновления данных без перезагрузки из JS скрипта -->*

<?php \yii\widgets\Pjax::*begin*(['id' => 'cart']);?>

<div class="cart-index">

<h1><?= Html::*encode*($this->title) ?></h1>

<?= GridView::*widget*([

'dataProvider' => $dataProvider,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

[

'label' => 'Наименование товара',

'format' => 'html',

'value' => function($model) {

return $model->product->name;

}

],

[

'label' => 'Фото товара',

'format' => 'html',

'value' => function($model) {

return Html::*img*('http://' . $\_SERVER["SERVER\_NAME"] . '/uploads/' . $model->product->photo,

['width' => 250, 'alt' => $model->product->name, 'title' => $model->product->name]);

}

],

'count',

[

'label' => 'Добавить',

'format' => 'raw',

'value' => function($model){

return '<button onclick="addCart('.$model->product->id.');" class="btn btn-success"> + </button>';

},

],

[

'label' => 'Удалить',

'format' => 'raw',

'value' => function($model){

return '<button onclick="removeCart('.$model->product->id.');" class="btn btn-success"> - </button>';

},

]

],

]); ?>

<?= Html::*input*('password','password', '',

['class' => 'form-control password mt-2', 'placeholder' => 'Введите ваш пароль']); ?>

<?= Html::*button*('Оформить заказ',

['class' => 'form-control btn btn-success mt-2', 'onclick' => 'addOrder();']); ?>

</div>

<?php \yii\widgets\Pjax::*end*()?>

Добавьте в файл web\js\my.js описание функций addCart() и removeCart(), которые вызываются при нажатии на кнопки 'Добавить' и 'Удалить'.

*//Функция добавления/увеличения товара в корзину*

function *addCart*(id) {

$.ajax({

method: "GET",

url: `/cart/create?product\_id=${id}`

}).done(function(message){

$('#info-message').html(message);

$.pjax.reload(

{

container:"#cart",

});

});

}

*//Функция удаления/уменьшения товара в корзине*

function *removeCart*(id) {

$.ajax({

method: "GET",

url: `/cart/delete?product\_id=${id}`

}).done(function(message){

$('#info-message').html(message);

$.pjax.reload(

{

container:"#cart",

});

});

}

В контроллере controllers\CartController.php измените метод actionCreate() отвечающий за добавление/увеличение товара в корзине и предусмотрите возможность проверки количества заказываемого товара (нельзя заказать больше, чем есть в наличии).

public function actionCreate($product\_id)

{

$product = \app\models\Product::*find*()->where(['id' => $product\_id])->andWhere(['>', 'count', '0'])->one();

if (!$product) {

return "Такого товара нет";

}

$itemInCart = Cart::*find*()->where(['product\_id' => $product\_id])->andWhere(['user\_id' => \Yii::*$app*->user->getId()])->one();

if (!$itemInCart){

$itemInCart = new Cart([

'product\_id' => $product\_id,

'user\_id' => \Yii::*$app*->user->getId(),

'count' => 1

]);

$itemInCart->save();

$product->count--;

$product->save(false);

return $product->name. " добавлен. Количество товара в корзине = $itemInCart->count";

} else {

if ($product->count == 0){

return "Нельзя больше добавить";

} else {

$product->count--;

$product->save(false);

$itemInCart->count++;

$itemInCart->save();

return $product->name. " добавлен. Количество товара в корзине = $itemInCart->count";

}

}

}

В контроллере controllers\CartController.php измените метод actionDelete() отвечающий за удаление/уменьшение товара из корзины.

public function actionDelete($product\_id)

{

$itemInCart = Cart::*find*()->where(['product\_id' => $product\_id])->andWhere(['user\_id' => \Yii::*$app*->user->getId()])->one();

if (!$itemInCart) {

return "Товар не найден";

}

$product = \app\models\Product::*find*()->where(['id' => $product\_id])->andWhere(['>', 'count', '0'])->one();

if ($itemInCart->count - 1 == 0){

$itemInCart->delete();

$product->count++;

$product->save(false);

return "Товар удален";

}

$product->count++;

$product->save(false);

$itemInCart->count--;

$itemInCart->save();

return $itemInCart->product->name . " уменьшено. Количество товара в корзине = $itemInCart->count";

}

## Оформление заказа

Добавьте в файл web\js\my.js описание функций addOrder (), который выполняет AJAX запрос на оформление заказа.

function *addOrder*() {

let password = $(".password");

if (!password.val()) {

$('#info-message').html("Укажите пароль");

return;

}

$.ajax({

method: "GET",

url: `/cart/addorder?password=${password.val()}`,

})

.done(function (msg) {

$('#info-message').html(msg);

$.pjax.reload({

container: '#cart'

});

});

}

В контроллере controllers\CartController.php добавьте метод actionAddorder () отвечающий за формирование заказа.

public function actionAddorder($password)

{

if (!\Yii::*$app*->user->identity->validatePassword($password)) {

return "Пароль не верный";

}

$order = new \app\models\Order([

'user\_id' => \Yii::*$app*->user->getId(),

'order\_status\_id' => \app\models\OrderStatus::*find*()->where(['code' => 'new'])->one()->id

]);

$order->save();

$itemInCart = \Yii::*$app*->user->identity->carts;

foreach ($itemInCart as $item) {

$itemInOrder = new \app\models\OrderList([

'order\_id' => $order->id,

'product\_id' => $item->product\_id,

'count' => $item->count,

'price' => $item->product->price \* $item->count,

]);

$itemInOrder->save();

$item->delete();

}

return "Заказ сформирован";

}

## Мои заказы

Разрешите просмотр заказов только авторизированным пользователям, для этого в контроллере controllers\OrderController.php измените метод behaviors().

*/\* Поведение. Доступ только для админа \*/*

public function behaviors()

{

return [

'access' => [

'class' => \yii\filters\AccessControl::class,

'only' => ['\*'],

'rules' => [

[

'actions' => ['index'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function($rule, $action){

return !\Yii::*$app*->user->isGuest;

}

],

[

'actions' => ['delete'],

'allow' => true,

'roles' => ['@'],

'matchCallback' => function ($rule, $action) {

$id = \Yii::*$app*->request->get('id');

$user\_id = \Yii::*$app*->user->identity->getId();

return Order::*find*()->where([

'user\_id' => $user\_id,

'id' => $id,

'order\_status\_id' => 1

])->exists();

}

],

],

],

];

}

Далее измените метод actionDelete(), который увеличивает кол-во товара на складе в следствии удаления заказа.

public function actionDelete($id)

{

$orderlist = \app\models\OrderList::*find*()->where(['order\_id' => $id])->all();

foreach ($orderlist as $item) {

$product = \app\models\Product::*find*()->where(['id' => $item->product\_id])->one();

$product->count += $item->count;

$product->save(false);

$item->delete();

}

$this->findModel($id)->delete();

return $this->redirect(['index']);

}

Русифицируйте метки в модели models\Order.php .

public function attributeLabels()

{

return [

'id' => 'ID',

'date' => 'Дата/Время заказа',

'rejection\_reason' => 'Причина отказа',

'user\_id' => 'User ID',

'order\_status\_id' => 'Статус заказа',

];

}

Измените модель поиска models\OrderSearch.php .

public function search($params)  
{  
 */\* Если администратор, показываем все заказы, в противном случае только заказы пользователя\*/* if (\Yii::*$app*->user->identity->isAdmin()) {  
 $query = Order::*find*()->orderBy(['id' => *SORT\_DESC*]);  
 } else {  
 $query = Order::*find*()->where(['user\_id' => \Yii::*$app*->user->identity->getId()])->orderBy(['id' => *SORT\_DESC*]);  
 }

Измените представление view\order\index.php .

<div class="order-index">

<h1><?= Html::*encode*($this->title) ?></h1>

*<!--*

*<p>*

<?= Html::*a*('Create Order', ['create'], ['class' => 'btn btn-success']) ?>

*</p>*

*-->*

<?php *// echo $this->render('\_search', ['model' => $searchModel]);* ?>

<?= GridView::*widget*([

'dataProvider' => $dataProvider,

'filterModel' => $searchModel,

'columns' => [

['class' => 'yii\grid\SerialColumn'],

'date',

'rejection\_reason',

[

'attribute' => 'order\_status\_id',

'label' => 'Статус заказа',

'format' => 'html',

'value' => function($model) {

return $model->orderStatus->name;

}

],

[

'label' => 'Список товаров',

'format' => 'html',

'value' => function($model) {

$orderlists = \app\models\OrderList::*find*()->where(['order\_id' => $model->id])->all();

$result ="";

foreach ($orderlists as $orderlist) {

$result .= "Товар: " . $orderlist->product->name . ". Кол-во: " . $orderlist->count . ". Цена: " . $orderlist->count \* $orderlist->price . "<br>";

}

return $result;

}

],

[

'label' => 'Удалить',

'format' => 'html',

'value' => function($model) {

if ($model->orderStatus->code == "new")

return \yii\helpers\Html::*a*('Удалить', Url::*toRoute*(['delete', 'id' => $model->id]));

},

'visible' => !Yii::*$app*->user->isGuest && !Yii::*$app*->user->identity->isAdmin()

],

[

'label' => 'Смена статуса',

'format' => 'html',

'value' => function($model) {

if ($model->orderStatus->code == "new")

return \yii\helpers\Html::*a*('Смена статуса', Url::*toRoute*(['update', 'id' => $model->id]));

},

'visible' => !Yii::*$app*->user->isGuest && Yii::*$app*->user->identity->isAdmin()

],

],

]); ?>

</div>

Измените представление view\order\\_form.php .

<div class="order-form">

<?php $form = ActiveForm::*begin*(); ?>

*<!--*

<?= $form->field($model, 'date')->textInput() ?>

<?= $form->field($model, 'user\_id')->textInput() ?>

*-->*

<?= $form->field($model, 'rejection\_reason')->textInput(['maxlength' => true]) ?>

<?= $form->field($model, 'order\_status\_id')->dropDownList(\yii\helpers\ArrayHelper::*map*(\app\models\OrderStatus::*find*()->all(), 'id', 'name')) ?>

<div class="form-group">

<?= Html::*submitButton*('Сохранить', ['class' => 'btn btn-success']) ?>

</div>

<?php ActiveForm::*end*(); ?>

</div>

Добавьте в ваш js скрипт для деактивации кнопки сохранить если выбран статус отказа заказа и не указана причина отказа.

$(document).ready( function () {

$('#order-rejection\_reason').change( function () {

if ( $('#order-rejection\_reason').val() == "" && $('#order-order\_status\_id').val() == 3 ){

$('button').attr('disabled', true);

} else {

$('button').removeAttr('disabled');

}

});

$('#order-order\_status\_id').change( function () {

if ( $('#order-rejection\_reason').val() == "" && $('#order-order\_status\_id').val() == 3 ){

$('button').attr('disabled', true);

} else {

$('button').removeAttr('disabled');

}

});

});

В контроллере controllers\OrderController.php измените метод actionUpdate ().

public function actionUpdate($id)

{

$model = $this->findModel($id);

if ($this->request->isPost && $model->load($this->request->post()) && $model->save()) {

*/\* Если статус отклонен, то увеличиваем кол-во товара на складе \*/*

$order = \app\models\Order::*find*()->where(['id' => $id])->one();

if ($order->orderStatus->code == "reject") {

$orderlist = \app\models\OrderList::*find*()->where(['order\_id' => $id])->all();

foreach ($orderlist as $item) {

$product = \app\models\Product::*find*()->where(['id' => $item->product\_id])->one();

$product->count += $item->count;

$product->save(false);

}

}

return $this->redirect(['index']);

}

return $this->render('update', [

'model' => $model,

]);

}