

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет
им. Г. И. Носова»
Многопрофильный колледж



**Методические указания
по подготовке к сдаче
демонстрационного экзамена
для обучающихся
специальности 09.02.07 Информационные системы
и программирование
квалификация: разработчик веб и мультимедийных приложений**

Магнитогорск, 2021

Предметно-цикловой комиссией
Информатики и вычислительной
техники

Председатель И.Г. Зорина
Протокол № 6 от 17.02.2021г.

Методической комиссией

Протокол №3 от «24» февраля
2021г.

Разработчик:

преподаватель ФГБОУ ВО «МГТУ им. Г.И. Носова» Многопрофильный
колледж

И.Г.Зорина

Методические указания разработаны на основе ФГОС СПО по специальности 09.02.07 Информационные системы и программирование, утвержденного приказом Министерства образования и науки Российской Федерации от 09.12.2016 г. № 1547, оценочных материалов для проведения демонстрационного экзамена КОД 09.02.07-3-2025: Разработчик веб и мультимедийных приложений.

Методические указания содержат общие положения по проведению демонстрационного экзамена, в полном объеме изложены рекомендации по выполнению заданий демонстрационного экзамена.

СОДЕРЖАНИЕ

1 ОБЩИЕ ПОЛОЖЕНИЯ	4
2 МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПОДГОТОВКЕ К ДЕМОНСТРАЦИОННОМУ ЭКЗАМЕНУ	16
3 ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ	27

1 ОБЩИЕ ПОЛОЖЕНИЯ

Демонстрационный экзамен направлен на определение уровня освоения выпускником материала, предусмотренного образовательной программой, и степени сформированности профессиональных умений и навыков путем проведения независимой экспертной оценки выполненных выпускником практических заданий в условиях реальных или смоделированных производственных процессов.

Демонстрационный экзамен направлен на контроль освоения следующих основных видов деятельности и соответствующих им общих и профессиональных компетенций:

Вид деятельности (вид профессиональной деятельности)	Перечень оцениваемых ОК, ПК	Перечень оцениваемых умений, навыков (практического опыта)
ИНВАРИАНТНАЯ ЧАСТЬ КОД		
Проектирование и разработка информационных систем	ПК: Разрабатывать подсистемы безопасности информационной системы в соответствии с техническим заданием	Умение: создавать и управлять проектом по разработке приложения и формулировать его задачи
		Умение: использовать языки структурного, объектно-ориентированного программирования и языка сценариев для создания независимых программ
		Умение: разрабатывать графический интерфейс приложения
		Практический опыт: модифицировать отдельные модули информационной системы
		Практический опыт: программировать в соответствии с требованиями технического задания
	ОК: Использовать современные	Умение: структурировать получаемую информацию

	средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности	
	ПК: Производить разработку модулей информационной системы в соответствии с техническим заданием	Умение: проектировать и разрабатывать систему по заданным требованиям и спецификациям Практический опыт: модифицировать отдельные модули информационной системы
Разработка дизайна веб-приложений	ПК: Разрабатывать дизайн-концепции веб-приложений в соответствии с корпоративным стилем заказчика	Умение: придерживаться оригинальной концепции дизайна проекта и улучшать его визуальную привлекательность
		Практический опыт: разрабатывать дизайн веб-приложений в соответствии со стандартами и требованиями заказчика
		Умение: Выбирать наиболее подходящее для целевого рынка дизайнерское решение
	ПК: Осуществлять разработку дизайна веб-приложения с учетом современных тенденций в области веб-разработки	Практический опыт: Создавать, использовать и оптимизировать изображения для веб-приложений
		Умение: Создавать «отзывчивый» дизайн, отображаемый корректно на различных устройствах и при разных разрешениях
		Умение: Использовать

		специальные графические редакторы
Проектирование, разработка и оптимизация веб-приложений	ПК: Разрабатывать веб-приложение в соответствии с техническим заданием	Практический опыт: выполнять верстку страниц веб-приложений
		Практический опыт: разрабатывать базы данных
		Умение: использовать открытые библиотеки (framework)
	ПК: Разрабатывать интерфейс пользователя веб-приложений в соответствии с техническим заданием	Умение: оформлять код программы в соответствии со стандартом кодирования
		Практический опыт: разрабатывать интерфейс пользователя
		Практический опыт: разрабатывать анимационные эффекты

Для проведения демонстрационного экзамена составляется расписание экзамена и консультаций.

Демонстрационный экзамен по специальности 09.02.07 Информационные системы и программирование проводится на профильном уровне.

Демонстрационный экзамен профильного уровня проводится по решению образовательной организации на основании заявлений выпускников на основе требований к результатам освоения образовательных программ среднего профессионального образования, установленных в соответствии с ФГОС СПО.

Комплект оценочной документации включает комплекс требований для проведения демонстрационного экзамена, перечень оборудования и оснащения, расходных материалов, средств обучения и воспитания, план застройки площадки демонстрационного экзамена, требования к составу экспертных групп, инструкции по технике безопасности, а также образцы заданий.

Задание демонстрационного экзамена включает комплексную практическую задачу, моделирующую профессиональную деятельность и

выполняемую в режиме реального времени.

5.2 Типовое задание для демонстрационного экзамена профильного уровня

5.2.1 Структура и содержание типового задания

Демонстрационный экзамен профильного уровня проводится с использованием единых оценочных материалов, включающих в себя конкретные комплекты оценочной документации (КОД), варианты заданий и критерии оценивания, разрабатываемых оператором. Комплект оценочной документации приведен в <https://bom.firpo.ru/file/public/77754/%D0%9A%D0%9E%D0%94%2009.02.07-3-2025%D0%A2%D0%BE%D0%BC%201.pdf> .

Задание состоит из 3-х модулей:

Модуль № 1: Проектирование и разработка информационных систем

Задание:

Разработать информационную систему для соответствующей предметной области.

Инструкция к выполнению практической части:

Для разработки используйте предоставленный сохраненный файл с базой данных. Разработайте минимально необходимый интерфейс для данной информационной системы. Используйте все знания в области языков разметки и программирования, но не забывайте что избыточность кода – плохое решение, а кодовое разнообразие – признак хорошего вкуса. Каждый раздел системы должен иметь свою отличительную черту.

Описание предметной области:

Портал клининговых услуг «Мой Не Сам» представляет собой информационную систему для формирования заявок на уборку жилых и производственных помещений. Перед началом использования портала пользователю необходимо пройти процедуру регистрации. В ходе регистрации он указывает данные о себе (ФИО, телефон, адрес электронной почты, логин и пароль).

Войдя в систему, заказчик может сформировать заявку на получение услуг, указав свой адрес, контактные данные и конкретный вид услуги из списка предоставляемых. Также заказчик может указать желаемую дату и время получения услуги, а также предпочтительный тип оплаты (наличными или с помощью банковской карты).

Заявки заказчиков хранятся в системе. В каждой заявке находятся данные заказчика, статус оказания услуги (новая заявка, услуга оказана или услуга отменена).

После подачи заявки администратор может подтвердить заявку, поставить отметку о выполнении или отклонить заявку с указанием причины отклонения.

Основной функционал информационной системы:

1. Страница регистрации. На данной странице необходимо предусмотреть добавление пользователя в систему. Пользователю необходимо предоставить возможность ввести логин, пароль, ФИО, телефон, адрес электронной почты. Все поля обязательны для заполнения. По кнопке «Зарегистрироваться» пользователь должен заноситься в базу.

2. Страница авторизации. На данной странице необходимо предусмотреть возможность ввода логина и пароля для зарегистрированных пользователей.

3. Страница создания заявки. На данной странице авторизованный пользователь имеет возможность просмотреть историю своих заявок, а также оставить новую заявку.

4. Страница формирования заявки. Пользователь указывает: свой адрес, контактные данные, желаемую дату и время получения услуги. Помимо этого, пользователь выбирает вид услуги из предоставленного списка (общий клининг, генеральная уборка, послестроительная уборка, химчистка ковров и мебели). Также пользователь указывает предпочтительный тип оплаты (наличные или банковская карта).

5. Панель администратора. Доступ в панель администратора осуществляется по логину adminka и паролю password. В панели администратора видны все заявки (ФИО заявителя, контактные данные и иные свойства заявки).

Модуль № 2: Разработка дизайна веб-приложений

Разработать информационную систему для соответствующей предметной области.

Инструкция к выполнению практической части:

Для разработки используйте предоставленный сохраненный файл с базой данных. Используйте все знания в области языков разметки и программирования, но не забывайте что избыточность кода-моветон, а кодовое разнообразие – признак хорошего вкуса. Каждый раздел системы

должен иметь свою отличительную черту. Не перегружайте вашу разработку обилием цветов, оставьте это для живописцев. При разработке поставьте себя на место потенциального клиента и постарайтесь создать идеальную информационную систему.

Вам необходимо также разработать дизайн всех страниц для использования со смартфоном с разрешением 390x844 рх. Дизайн можно представить в виде файлов изображений .png (отдельное изображение для каждой страницы), либо в виде .html файлов (отдельный файл для каждой страницы).

Интегрировать дизайн в разрабатываемую информационную систему не требуется.

Описание предметной области:

Портал клининговых услуг «Мой Не Сам» представляет собой информационную систему для формирования заявок на уборку жилых и производственных помещений. Перед началом использования портала пользователю необходимо пройти процедуру регистрации. В ходе регистрации он указывает данные о себе (ФИО, телефон, адрес электронной почты, логин и пароль).

Войдя в систему, заказчик может сформировать заявку на получение услуг, указав свой адрес, контактные данные и конкретный вид услуги из списка предоставляемых. Также заказчик может указать желаемую дату и время получения услуги, а также предпочтительный тип оплаты (наличными или с помощью банковской карты).

Заявки заказчиков хранятся в системе. В каждой заявке находятся данные заказчика, статус оказания услуги (новая заявка, услуга оказана или услуга отменена).

После подачи заявки администратор может подтвердить заявку, поставить отметку о выполнении или отклонить заявку с указанием причины отклонения.

Основной функционал информационной системы:

1. Страница регистрации. На данной странице необходимо предусмотреть добавление пользователя в систему. Пользователю необходимо предоставить возможность ввести уникальный логин, пароль, ФИО, телефон, адрес электронной почты. Все поля обязательны для заполнения.

По кнопке «Зарегистрироваться» пользователь должен заноситься в базу.

2. Страница авторизации. На данной странице необходимо предусмотреть возможность ввода логина и пароля для зарегистрированных пользователей. Попытки некорректного ввода логина и пароля должны сопровождаться сообщениями.

3. Страница создания заявки. На данной странице авторизованный пользователь имеет возможность просмотреть историю своих заявок, а также оставить новую заявку.

4. Страница формирования заявки. Пользователь указывает: свой адрес, контактные данные, желаемую дату и время получения услуги. Помимо этого, пользователь выбирает вид услуги из предоставленного списка (общий клининг, генеральная уборка, послестроительная уборка, химчистка ковров и мебели). Также пользователь указывает предпочтительный тип оплаты (наличные или банковская карта). Все поля обязательны для заполнения.

5. Панель администратора. Доступ в панель администратора осуществляется по логину adminka и паролю password. В панели администратора видны все заявки (ФИО заявителя, контактные данные и иные свойства заявки).

Администратор может сменить статус на «в работе», «выполнено» или «отменено» (обязательно с указанием причины отмены).

Модуль № 3: Проектирование, разработка и оптимизация веб-приложений

Разработать информационную систему для соответствующей предметной области.

Инструкция к выполнению практической части:

Необходимо разработать базу данных, позволяющую реализовать функционал разрабатываемой информационной системы с учетом особенностей предметной области. В качестве документации к системе разработайте диаграмму базы данных с помощью средств графических редакторов. Используйте все знания в области языков разметки и программирования, но не забывайте что избыточность кода-моветон, а кодовое разнообразие – признак хорошего вкуса. Каждый раздел системы должен иметь свою отличительную черту. Для удобства разработки можете воспользоваться фреймворком. Не перегружайте вашу разработку обилием цветов, оставьте это для живописцев. При разработке поставьте себя на место потенциального клиента и постарайтесь создать идеальную информационную систему, как с точки зрения пользователя, так и с

точки зрения разработчика. Помните, что пользователи разные. Так же как и устройства, с которых они осуществляют вход в Интернет.

Вам необходимо также разработать дизайн всех страниц для использования со смартфоном с разрешением 390x844 px. Дизайн можно представить в виде файлов изображений .png (отдельное изображение для каждой страницы), либо в виде .html файлов (отдельный файл для каждой страницы).

Интегрируйте Ваш дизайн в разрабатываемую информационную систему. Предусмотрите анимацию для улучшения пользовательского опыта.

Описание предметной области:

Портал клининговых услуг «Мой Не Сам» представляет собой информационную систему для формирования заявок на уборку жилых и производственных помещений. Перед началом использования портала пользователю необходимо пройти процедуру регистрации. В ходе регистрации он указывает данные о себе (ФИО, телефон, адрес электронной почты, логин и пароль) (логины разных пользователей не должны совпадать).

Войдя в систему, заказчик может сформировать заявку на получение услуг, указав свой адрес, контактные данные и конкретный вид услуги из списка предоставляемых. Также заказчик может указать желаемую дату и время получения услуги, а также предпочтительный тип оплаты (наличными или с помощью банковской карты).

Заявки заказчиков хранятся в системе. В каждой заявке находятся данные заказчика, статус оказания услуги (новая заявка, услуга оказана или услуга отменена).

После подачи заявки администратор может подтвердить заявку, поставить отметку о выполнении или отклонить заявку с указанием причины отклонения.

Основной функционал информационной системы:

1. Страница регистрации. На данной странице необходимо предусмотреть добавление пользователя в систему. Пользователю необходимо предоставить возможность ввести уникальный логин, пароль (минимум 6 символов), ФИО (символы кириллицы и пробелы), телефон (в формате +7(XXX)-XXX-XX-XX), адрес электронной почты (формат электронной почты). Все поля обязательны для заполнения. Ошибки валидации должны отображаться на форме. По кнопке

«Зарегистрироваться» пользователь должен заноситься в базу, если поля прошли валидацию.

2. Страница авторизации. На данной странице необходимо предусмотреть возможность ввода логина и пароля для зарегистрированных пользователей. Попытки некорректного ввода логина и пароля должны сопровождаться сообщениями.

3. Страница создания заявки. На данной странице авторизованный пользователь имеет возможность просмотреть историю своих заявок, а также оставить новую заявку.

4. Страница формирования заявки. Пользователь указывает: свой адрес, контактные данные (номер телефона в формате +7(XXX)-XXX-XX-XX), желаемую дату и время получения услуги. Помимо этого, пользователь выбирает вид услуги из предоставленного списка (общий клининг, генеральная уборка, послестроительная уборка, химчистка ковров и мебели). Если требуемый вид услуги отсутствует в списке, пользователь отмечает соответствующий чекбокс («Иная услуга»), после чего ему становится доступно поле для ввода текстовой информации, где он описывает, какая конкретно услуга ему необходима. Также пользователь указывает предпочтительный тип оплаты (наличные или банковская карта). Все поля обязательны для заполнения.

5. Панель администратора. Доступ в панель администратора осуществляется по логину adminka и паролю password. В панели администратора видны все заявки (ФИО заявителя, контактные данные и иные свойства заявки).

Администратор может сменить статус на «в работе», «выполнено» или «отменено» (обязательно с указанием причины отмены)

5.2.2 Оснащение рабочего места для проведения демонстрационного экзамена по типовому заданию

Материально-техническая база соответствует инфраструктурному листу КОД 09.02.07-3-2025: Разработчик веб и мультимедийных приложений

5.3 Критерии оценки выполнения задания демонстрационного экзамена

Процедура оценивания результатов выполнения заданий демонстрационного экзамена осуществляется членами экспертной группы по 100-балльной системе в соответствии с требованиями

комплекта оценочной документации.

Распределение баллов по критериям оценивания демонстрационного экзамена профильного уровня представлен в таблице.

№ п/п	Модуль задания (вид деятельности, вид профессиональной деятельности)	Критерий оценивания	Баллы
1	Проектирование и разработка информационных систем	Разработка подсистемы безопасности информационной системы в соответствии с техническим заданием	18,00
		Проведение разработки модулей информационной системы в соответствии с техническим заданием	4,00
		Использование современных средств поиска, анализа и интерпретации информации и информационных технологий для выполнения задач профессиональной деятельности	4,00
2	Разработка дизайна вебприложений	Разработка дизайн-концепций вебприложений в соответствии с корпоративным стилем заказчика	8,00
		Формирование требований к дизайну веб-приложений на основе анализа предметной области и целевой аудитории	8,00
		Осуществление разработки дизайна веб-приложения с учетом современных тенденций в области веб-разработки	12,00
3	Проектирование, разработка и оптимизация вебприложений	Разработка веб-приложения в соответствии с техническим заданием	14,00
		Разработка интерфейса	12,00

		пользователя веб-приложений в соответствии с техническим заданием	
ИТОГО			80,00

Необходимо осуществить перевод количества баллов в оценки «отлично», «хорошо», «удовлетворительно», «неудовлетворительно». Перевод полученного количества баллов в оценки осуществляется государственной экзаменационной комиссией с обязательным присутствием главного эксперта.

Перевод баллов в оценку может быть осуществлен на основе таблицы:

Оценка ГИА	«2»	«3»	«4»
Отношение полученного количества баллов к максимально возможному (в процентах)	0,00 - 19,99%	20,00 – 39,99%	40,00 – 69,99%

Баллы выставляются в протоколе проведения демонстрационного экзамена, который подписывается каждым членом экспертной группы и утверждается главным экспертом после завершения экзамена для экзаменационной группы.

При выставлении баллов присутствует член ГЭК, не входящий в экспертную группу, присутствие других лиц запрещено.

Подписанный членами экспертной группы и утвержденный главным экспертом протокол проведения демонстрационного экзамена далее передается в ГЭК для выставления оценок по итогам ГИА.

Оригинал протокола проведения демонстрационного экзамена передается на хранение в образовательную организацию в составе архивных документов.

Статус победителя, призера чемпионатов профессионального мастерства, проведенных Агентством (Союзом «Агентство развития профессиональных сообществ и рабочих кадров «Молодые профессионалы (Ворлдскиллс Россия)») либо международной организацией «WorldSkills International», в том числе «WorldSkills Europe» и «WorldSkills Asia», и участника национальной сборной России

по профессиональному мастерству по стандартам «Ворлдскиллс» выпускника по профилю осваиваемой образовательной программы среднего профессионального образования засчитывается в качестве оценки «отлично» по демонстрационному экзамену в рамках проведения ГИА по данной образовательной программе среднего профессионального образования.

2 МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПОДГОТОВКЕ К ДЕМОНСТРАЦИОННОМУ ЭКЗАМЕНУ

Модуль 1. Проектирование и разработка информационных систем

Модуль 2. Разработка дизайна веб-приложений

Модуль 3. Проектирование, разработка и оптимизация веб-приложений

2.1 Установка и настройка Laravel

Перед установкой проверьте, что в командной строке работают команды `php` и `composer`, т.к. они нужны для установки фреймворка.

Если все работает, то перейдите в командной строке в корневую папку с веб-сервером:

Далее необходимо установить фреймворк используя `Composer`:

```
composer create-project laravel/laravel project
```

Команда содержит следующие параметры:

- `composer` – программа, которую запускаем;
- `create-project` – создание проекта;
- `prefer-dist` – параметр при котором `Composer` будет скачивать стабильные, запакованные версии проекта, вместо клонирования из системы контроля версий (что значительно медленнее);
- `laravel/laravel` – название пакета;
- `project` – название папки, которая будет создана. Если вы хотите установить пакет в текущую папку, то используйте точку, которая означает текущую директорию.

В результате выполнения данной команды в текущей папке будет создана папка `project`, а в ней будет находиться наше приложение.

Далее необходимо убедиться, что сервер запущен и после этого можно переходить по адресу `http:// project /`, где `project` - это та папка, в которую поставили фреймворк при его установке.

В результате должны увидеть список папок и файлов.

После установки необходимо проверить, что приложение работает. Для этого нужно понять, как и что запускать. Корневой точкой входа в приложение является папка `public` и именно в ней находится файл `index.php`, который и запускает работу всего приложения.

Для того чтобы увидеть фреймворк в действии нам потребуется браузер. Откройте ваш сайт в браузере по адресу `http://project/public`

Далее нужно сконфигурировать настройки веб-сервера таким образом, чтобы `DocumentRoot` указывал сразу на папку `public`. Для этого нужно в корне приложения создать файл, у которого не будет имени, а

будет только расширение .htaccess. Это файл настройки Apache. Откроем его в текстовом редакторе и напишем следующий код:

```
RewriteEngine on
RewriteRule ^(.*)$ public/$1
```

RewriteEngine on – это включить перенаправления.

RewriteRule задает правила перенаправления (что и куда перенаправляется).

2.2 Основы баз данных

База данных – это программа, которая хранит большие объемы связанной информации и позволяет производить поиск по этим данным.

Реляционные базы данных состоят из таблиц, где и хранится вся информация. При создании базы данных нужно сразу определиться с тем какие таблицы будут в базе данных. Хорошо спроектированная база данных позволит легко обслуживать базу данных в будущем, а также обеспечит целостность данных.

Каждая таблица имеет название, столбцы и строки. Столбцы –это свойства, а строки – записи.

При этом в каждой таблице должен быть первичный ключ - одно или несколько полей, которые образуют уникальное значение, которое никогда не повторится в рамках данной таблицы.

Чаще всего в роли первичного ключа выступает столбец id - идентификатор.

Нормализация

Представим типичную ситуацию с Блогом, когда нам нужно хранить пользовательские посты. О постах нам известно следующее:

- У постов есть название;
- У постов есть текст;
- Пост написан каким-то пользователем;
- Нужно сохранить дату и время написания поста.

В таком случае, мы можем представить вот такую таблицу:

id	user	title	text	created_at	updated_at

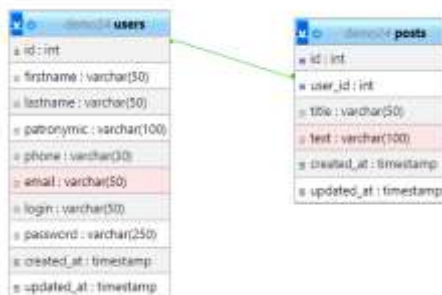
Однако в данной таблице есть проблема - она не нормализована! Это значит, что данные собраны не оптимально и дублируются. Например, посты и пользователи могут повторяться и не один раз.

Давайте представим, что у нас есть четыре пользователя. Эти пользователи могут написать более 5000 постов, а потом кто-нибудь решит изменить свое имя. В результате нам придется изменить все упоминания о изменяемом имени во всех постах, а может быть не только

в постах.

Такая ситуация очень опасна, особенно если имена пользователей используются не только в постах – тогда где-то может остаться не измененное имя пользователя!

Чтобы такого не было необходимо разделять данные по разным таблицам – пользователи отдельно и посты отдельно!



Теперь у нас вместо одной таблицы есть две:

- Таблица пользователей;
- Таблица постов.

При этом данные в таблице пользователей не будут дублироваться! В них будет находиться ровно столько записей, сколько существует пользователей.

В таблице постов мы будем лишь ссылаться на идентификаторы используемых пользователей.

2.3 Работа с миграциями

Подключение к базе данных

Перед тем как создавать миграции необходимо настроить подключение к базе данных. Подключение к базе данных настраивается в файле `.env`, который находится в корне приложения.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=database
DB_USERNAME=username
DB_PASSWORD=password
```

Вместо `database` нужно подставить имя базы данных, вместо `username` и `password` логин и пароль соответственно.

Миграции

Миграции в приложении находятся в папке

database/migrations и по умолчанию там уже созданы 4 миграции:

1. `create_users_table` – таблица пользователей;
2. `create_password_resets_table` – таблица для восстановления паролей;
3. `create_failed_jobs_table` – таблица задач;
4. `create_personal_access_tokens_table` – таблица для токенов авторизации.

Создание миграции

Для создания миграций в Laravel можно воспользоваться командой Artisan:

```
php artisan make:migration <migration_name>
```

Вместо `<migration_name>` нужно указать название миграции. Название должно быть написано по правилу `create_НАЗВАНИЕ-ТАБЛИЦЫ_table`.

Структура миграции

По умолчанию любая миграция – это класс, который содержит 2 метода `up` и `down`.

Метод `up` вызывается при применении миграции, а `down` при ее откате.

Если в методе `up` будет реализовано создание таблицы, то в методе `down` необходимо реализовать обратное действие, т.е. ее удаление.

Статический метод `create` из класса `Schema` принимает 2 параметра – первый это имя создаваемой таблицы, а второй параметр – это анонимная функция, где описывается создаваемая таблица.

Методы создания полей

Идентификатор

Для создания поля с первичным ключом необходимо использовать метод `id`.

Данный метод создаст в таблице большое не отрицательное целое число с названием `id`.

```
$table->id();
```

Timestamps

При создании миграции там уже есть метод `timestamps` - этот метод создает в таблице два поля:

- Временную метку `created_at` – в данном поле будет храниться дата и время создания записи;
- Временную метку `updated_at` – в данном поле будет храниться дата и время обновления записи.

При этом приложение самостоятельно будет заполнять данные поля при создании записи или при обновлении.

Строка

Для создания текстового поля в таблице необходимо использовать метод `string`.

Данный метод принимает два параметра - название поля и его длина, причем длина является не обязательной и по умолчанию составляет 255.

```
$table->string('name');
```

Числа

В базе данных MySQL для работы с числами есть несколько методов: `tinyInteger`, `smallInteger`, `mediumInteger`, `integer`, `bigInteger`.

Все методы принимают 3 параметра:

1. название поля в базе – обязательный параметр;
2. будет ли поле увеличиваться как автоинкремент – не обязательный параметр, `true/false`;
3. будет ли поле не отрицательным – не обязательный параметр, `true/false`.

```
$table->tinyInteger('age');
$table->integer('code');
```

Булевый тип

Поле булевого типа можно создать с помощью метода `boolean`, который принимает только 1 обязательный параметр – название поля.

Дата

Для добавления поля с типом даты необходимо вызвать метод `date` и передать туда название поля.

Время

Для добавления поля с типом времени необходимо вызвать метод `time` и передать туда название поля.

Перечисление

Чтобы добавить поле, которое может принимать только определенные значения из списка нужно вызвать метод `enum`, который принимает два обязательных параметра – название поля и массив допустимых значений.

```
$table->enum('gender', ['М', 'Ж']);
```

Временные метки

Для хранения временных меток необходимо использовать метод `timestemp`.

Модификаторы

Не обязательное поле

Если поле не обязательно для заполнения при создании записи, то можно использовать модификатор `nullable`, который разрешает хранить `NULL` в поле.

```
$table->string('token')->nullable();
```

Значение по умолчанию

С помощью модификатора `default` можно задать значение по умолчанию, и оно будет использоваться при создании записи если в запросе не было указано явное значение.

```
$table->tinyInteger('role')->default(1);  
$table->boolean('read')->default(false);
```

Не отрицательное

Модификатор `unsigned` указывает на то, что поле будет хранить не отрицательных значения. Данный модификатор может быть использован только для числовых полей, таких как `tinyInteger`, `integer` и т.д.

```
$table->tinyInteger('code')->unsigned();
```

Связи

Для создания связи можно использовать метод `foreignId` в связке с методом `constrained` как в примере ниже:

```
public function up()
{
    Schema::create('posts', function (Blueprint
$table) {
        $table->id();
        $table->string('title');
        $table->text('text');
        $table->foreignId('user_id')->
constrained('users')->cascadeOnDelete();
        $table->timestamps();
    });
}
```

Метод `foreignId` создает большое не отрицательное целое число, а также внешний ключ. Метод `constrained` указывает на таблицу с которой будет связь. По умолчанию связь будет с полем `id` во внешней таблице. Метод `cascadeOnDelete` задает каскадное удаление данных.

Работа с миграциями

Для применения миграций необходимо использовать Artisan команду `migrate`.

Данная команда применит все миграции, которые еще не были исполнены.

```
php artisan migrate
```

Для отката последней миграции необходимо использовать команду Artisan `migrate:rollback`.

Данная команда отменит последние миграции.

```
php artisan migrate:rollback
```

Если в процессе разработки необходимо применить все миграции

заново, то можно использовать команду `migrate:fresh`.

```
php artisan migrate:fresh
```

2.4 Маршруты и контроллеры

Маршруты

В Laravel за маршрутизацию отвечают файлы `web.php` и `api.php` из папки `routes`.

Файл `web.php` отвечает за маршрутизацию интерфейсной части, а `api.php` за маршрутизацию запросов к API.

За маршрутизацию в Laravel отвечает класс `Route`. Чтобы определить маршрут нужно вызвать один из статических методов класса `Route`:

```
Route::get('/users', ...); // Будет обрабатывать
GET-маршрут с адресом /users
Route::post('/users', ...); // Будет обрабатывать
POST-маршрут с адресом /users
Route::patch('/users', ...); // Будет обрабатывать
PATCH-маршрут с адресом /users
Route::put('/users', ...); // Будет обрабатывать
PUT-маршрут с адресом /users
Route::delete('/users', ...); // Будет обрабатывать
DELETE-маршрут с адресом /users
```

В примере мы обращаемся к классу `Route` и вызываем статические методы `get`, `post`, `patch`, `put` или `delete` для обработки запроса соответствующего типа.

Все эти методы принимают два параметра:

1. **URI** – строка с обрабатываемым адресом. В Laravel и REST API принято использовать наименование ресурсов. Ресурсы – это сущности с которыми мы работаем. Например, если мы хотим управлять Книгами, то это ресурс `Book` и адрес будет `/books`.

2. **Action** – действие-обработчик, которое может быть функцией, строкой или массивом.

Самый простой способ обработать какой-либо маршрут – это передать анонимную функцию в качестве второго параметра:

```
Route::get('/users', function() {
    return ['Пользователь 1', 'Пользователь 2'];
});
```

Такая функция обязательно должна что-то возвращать с помощью ключевого слова `return`. Все что будет возвращено из этой функции будет передано клиенту в качестве ответа на его запрос.

Такой способ иногда можно использовать, например для обработки простого аякс-запроса при условии, что их у вас не много и в целом приложение не большое. Однако в большом приложении или при разработке API такой способ не рекомендуется, т.к. каждая функция будет довольно большая и кода в файле с маршрутами будет очень много!

Контроллеры

Контроллеры позволяют разделить код по логике, связанной с ресурсами и вынести все из `api.php`.

Контроллеры в Laravel находятся в папке `app/Http/Controllers`.

Создать новый контроллер можно с помощью Artisan команды:

```
php artisan make:controller BookController
```

В результате будет создан файл `BookController.php` в папке `app/Http/Controllers`:

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
class BookController extends Controller
{
    //
}
```

Контроллер – это просто класс, который наследуется от класса `Controller`.

Далее мы можем создать нужные нам методы в этом классе:

```
<?php
namespace App\Http\Controllers;
class BookController extends Controller
{
```



```

public function index()
{
    return 'books';
}
}

```

Теперь мы можем избавиться от анонимной функции в web.php и вместо нее указать строку с адресом контроллера и метода, который будет обрабатывать данный маршрут в формате **адрес_контроллера@метод_в_контроллере**

```

Route::get('/books',
'\App\Http\Controllers\BookController@index');

```

Теперь при запросе на адрес /api/books должен вызываться метод index из контроллера BookController.

Если вы не хотите прописывать адрес до контроллера в виде строки, то стоит использовать массив:

```

Route::get('/books', [BookController::class,
'index']);

```

Теперь вторым параметром мы передаем массив состоящих из 2 элементов:

- полное имя класса;
- название метода в контроллере.

Методы

При работе с ресурсами в laravel принято использовать следующие названия методов в контроллере:

Название метода контроллера	Его роль (HTML)	Его роль (API)
index	Возвращает страницу со всеми записями	Возвращает все записи
show	Возвращает страницу с одной записью	Возвращает одну запись
create	Возвращает страницу с формой создания новой записи	-
store	Сохраняет запись в базу данных	Сохраняет запись в базу данных

edit	Возвращает страницу с формой редактирования существующей записи	-
update	Обновляет запись в базе данных	Обновляет запись в базе данных
destroy	Удаляет запись из базы данных	Удаляет запись из базы данных

Наименования маршрутов

Каждому маршруту можно задать название с помощью функции `name`.

```
Route::get('/users', [UserController::class, 'index'])->name('users.index');
```

При работе с API вам это не пригодится, а вот при работе с представлениями без этого никуда!

Ресурсы

Если вам необходимо реализовать все CRUD операции для управления ресурсом с помощью страниц, то вы можете воспользоваться методом `resource`, а не писать маршруты для каждого действия по отдельности:

```
Route::resource('books', BookController::class);
```

В результате данный метод создаст все необходимые маршруты для управления ресурсом `Books`:

- GET /books - страница со всеми ресурсами (книгами);
- GET /books/create - страница создания нового ресурса (книги);
- POST /books - создание нового ресурса (книги);
- GET /books/{book} - страница с одним ресурсом (книгой) по ID;
- GET /books/{book}/edit - страница редактирования одного ресурса (книги) по ID;
- PUT|PATCH /books/{book} - изменение одного ресурса (книги) по ID;
- DELETE /books/{book} - удаление одного ресурса (книги) по ID.

Также все созданные маршруты будут подписаны в соответствии с именем метода в контроллере.

3 ИНФОРМАЦИОННО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

Основные источники

1. Варфоломеева, А. О. Информационные системы предприятия [Электронный ресурс] : учебное пособие / А. О. Варфоломеева, А. В. Коряковский, В. П. Романов. — 2-е изд., перераб. и доп. — Москва : ИНФРА-М, 2019. — 330 с. — (Среднее профессиональное образование). - Режим доступа: <https://znanium.com/read?id=335060> — Загл. с экрана.
2. Гагарина, Л. Г. Разработка и эксплуатация автоматизированных информационных систем [Электронный ресурс] : учебное пособие / Л. Г. Гагарина. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2019. — 384 с. — (Среднее профессиональное образование). - Режим доступа: <https://znanium.com/read?id=333679> — Загл. с экрана.
3. Коваленко, В. В. Проектирование информационных систем [Электронный ресурс] : учебное пособие / В. В. Коваленко. — Москва : ФОРУМ : ИНФРА-М, 2021. — 320 с. — Режим доступа: <https://znanium.com/catalog/document?id=390037> — Загл. с экрана.
4. Исаев, Г. Н. Управление качеством информационных систем [Электронный ресурс] : учебное пособие / Исаев Г. Н. - Москва : НИЦ ИНФРА-М, 2022. - 248 с.: 60х90 1/16. - (Высшее образование: Бакалавриат) (Переплёт 7БЦ) ISBN 978-5-16-011794-2 - Режим доступа: <https://znanium.com/read?id=393205> — Загл. с экрана.
5. Немцова, Т.И. Компьютерная графика и web-дизайн [Электронный ресурс]: учебное пособие / Т.И. Немцова, Т.В. Казанкова, А.В. Шнякин; под ред. Л.Г. Гагариной. — Москва: ИД «ФОРУМ»: ИНФРА-М, 2019. — 400 с. + Доп. материалы. - Режим доступа: <https://new.znanium.com/read?id=329728> — (Среднее профессиональное образование).
6. Немцова, Т. И. Практикум по информатике. Компьютерная графика и web-дизайн: учебное пособие / Т.И. Немцова, Ю.В. Назарова; под ред. Л.Г. Гагариной. — Москва: ФОРУМ: ИНФРА-М, 2023. — 288 с. + Доп. материалы [Электронный ресурс]. — (Среднее профессиональное образование). - ISBN 978-5-8199-0800-6. - Режим доступа: <https://znanium.com/read?id=428047>
7. Лисьев, Г. А. Программное обеспечение компьютерных сетей и web-серверов [Электронный ресурс] : учебное пособие / Г. А. Лисьев, П. Ю. Романов, Ю. И. Аскерко. — Москва : ИНФРА-М, 2021. — 145 с. — (Среднее профессиональное образование). - Режим доступа: <https://znanium.com/read?id=365037> . — Загл. с экрана.

8. Немцова, Т. И. Компьютерная графика и web-дизайн [Электронный ресурс] : учебное пособие / Т. И. Немцова, Т. В. Казанкова, А. В. Шнякин ; под ред. Л.Г. Гагариной. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2022. — 400 с. + Доп. материалы. - Режим доступа: <https://znanium.com/read?id=379822> — (Среднее профессиональное образование). — Загл. с экрана.
9. Цупин, В. А. Управление контентом. Практикум [Электронный ресурс] : учебное пособие / В.А. Цупин, М.М. Ниматулаев. — Москва : ИНФРА-М, 2022. — 211 с. — (Высшее образование: Бакалавриат). — Режим доступа: <https://znanium.com/read?id=379146> . — Загл. с экран.
10. Шаньгин, В. Ф. Комплексная защита информации в корпоративных системах [Электронный ресурс] : учебное пособие / В.Ф. Шаньгин. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2022. — 592 с. — (Высшее образование: Бакалавриат). - Режим доступа: <https://znanium.com/read?id=389857> . — Загл. с экрана.

Дополнительные источники

1. Зараменских, Е. П. Информационные системы: управление жизненным циклом : учебник и практикум для среднего профессионального образования / Е. П. Зараменских. — Москва : Издательство Юрайт, 2023. — 431 с. — (Профессиональное образование). — ISBN 978-5-534-11624-3. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/518514> – Загл. с экрана.
2. Сысоева, Л. А. Управление проектами информационных систем [Электронный ресурс] : учебное пособие / Л. А. Сысоева, А. Е. Сатунина. — Москва : ИНФРА-М, 2019. — 345 с. — (Высшее образование: Бакалавриат). — Режим доступа: <https://znanium.com/read?id=342011> – Загл. с экрана.
3. Павловская, Е.Э. Графический дизайн. Современные концепции [Электронный ресурс]: учебное пособие для вузов / Е. Э. Павловская [и др.]; ответственный редактор Е.Э. Павловская. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2019. — 119 с. — (Университеты России). — ISBN 978-5-534-11169-9. — Режим доступа: <https://www.biblio-online.ru/bcode/444790>
4. Лаврентьев, А.Н. Цифровые технологии в дизайне. История, теория, практика [Электронный ресурс]: учебник и практикум для вузов / А. Н. Лаврентьев [и др.]; под редакцией А.Н. Лаврентьева. - 2-е изд., испр. и доп. - Москва: Издательство Юрайт, 2019. — 208 с. —

- (Авторский учебник). — ISBN 978-5-534-07962-3. — Режим доступа: <https://www.biblio-online.ru/bcode/424029>
5. Шаньгин, В. Ф. Информационная безопасность компьютерных систем и сетей [Электронный ресурс] : учебное пособие / В. Ф. Шаньгин. — Москва : ИД «ФОРУМ» : ИНФРА-М, 2021. — 416 с. — (Среднее профессиональное образование). - Режим доступа: <https://znanium.com/read?id=336332>. — Загл. с экрана.

Интернет-ресурсы

1. Интуит Национальный открытый университет курс. Проектирование информационных систем https://www.intuit.ru/studies/professional_retraining/14629/video_courses/330/info
2. Интуит Национальный открытый университет курс Управление развитием информационных систем https://www.intuit.ru/studies/professional_retraining/14629/courses/388/info
3. Интуит - Национальный открытый университет. Курс «HTML5. Основы клиентской разработки» [Электронный ресурс]. — Режим доступа: https://www.intuit.ru/studies/professional_retraining/16256/courses/976/info, свободный. — Загл. с экрана. Яз. рус.
4. Интуит - Национальный открытый университет. Курс «Создание компьютерной анимации в Adobe Flash CS3 Professional» [Электронный ресурс]. — Режим доступа: https://www.intuit.ru/studies/professional_retraining/961/courses/375/info, свободный. — Загл. с экрана. Яз. рус.
5. Курсы по программированию: [Электронный ресурс] // URL: <https://htmlacademy.ru/>
6. Интуит — национальный открытый университет. Основы поисковой оптимизации (SEO) [Электронный ресурс]. — Режим доступа: https://www.intuit.ru/studies/professional_retraining/16256/video_courses/1121/info, свободный. — Загл. с экрана. Яз. рус.
7. Интуит — национальный открытый университет. Основы работы с Google Analytics [Электронный ресурс]. — Режим доступа: <https://www.intuit.ru/studies/courses/4493/1018/info>, свободный. — Загл. с экрана. Яз. рус.
8. Интуит — национальный открытый университет. Основы клиентской оптимизации [Электронный ресурс]. — Режим доступа: <https://www.intuit.ru/studies/courses/1179/333/info>, свободный. — Загл. с экрана. Яз. рус.

9. Интуит – национальный открытый университет. Основы программирования на AJAX [Электронный ресурс]. – Режим доступа:
https://www.intuit.ru/studies/professional_retraining/16256/video_courses/1017/info, свободный. – Загл. с экрана. Яз. рус.
10. Портал по php, MySQL и другим веб-технологиям [Электронный ресурс] - Режим доступа : <http://www.php.su>