Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Курганский государственный университет»

Кафедра «Программное обеспечение автоматизированных систем»

УТВЕРЖДАЮ:
Первый проректор
_____ Т. Р. Змызгова
«02» сентября 2022 г.

Рабочая программа учебной дисциплины

ТЕХНОЛОГИИ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЙ

образовательной программы высшего образования –
программы бакалавриата

09.03.03 Прикладная информатика
направленность
Интеллектуальные информационные системы и технологии

формы обучения – очная и заочная

Курган 2022

Рабочая программа дисциплины «Технологии разработки web-приложений» составлена в соответствии с учебными планами по программе бакалавриата «Прикладная информатика» (Интеллектуальные информационные системы и технологии), утвержденными для очной формы обучения «30» августа 2022 года, для заочной формы обучения «30» августа 2022 года.

Рабочая программа дисциплины одобрена на заседании кафедры «Программное обеспечение автоматизированных систем» «1» сентября 2022 года, протокол № 1.

Рабочую программу составил:

Доцент кафедры
«Программное обеспечение
автоматизированных систем»
к.т.н., доцент _____ А.М. Семахин

Заведующий кафедрой
«Программное обеспечение
автоматизированных систем»
к.т.н., доцент _____ В. К. Волк

Согласовано:

Начальник
Управления
образовательной деятельности _____ И. В. Григоренко

Специалист
по учебно-методической работе
Учебно-методического отдела _____ Г.В. Казанкова

1. ОБЪЕМ ДИСЦИПЛИНЫ

Всего: 7 зачетных единиц трудоемкости (252 академических часа)

Очная форма обучения

Вид учебной работы	На всю дисциплину	Семестр	
		4	5
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**112**	**64**	**48**
Лекции	32	16	16
Лабораторные работы	64	32	32
Практические занятия	16	16	-
Аудиторные занятия в интерактивной форме, часов	-	-	
Самостоятельная работа, всего часов в том числе:	**140**	**44**	**96**
Контрольная работа	18	18	-
Курсовой проект	36	-	36
Подготовка к зачёту	18	18	-
Подготовка к экзамену	27	-	27
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплны)	41	8	33
Вид промежуточной аттестации	**зачёт**	**зачёт**	**-**
	экзамен	**-**	**экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**252**	**108**	**144**

Заочная форма обучения

Вид учебной работы	На всю дисциплину	Семестр	
		5	6
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**22**	**12**	**10**
Лекции	6	6	-
Лабораторные работы	12	6	6
Практические занятия	4	-	4
Аудиторные занятия в интерактивной форме, часов	-	-	
Самостоятельная работа, всего часов в том числе:	**230**	**96**	**134**
Контрольная работа	18	18	-
Курсовой проект	36	-	36
Подготовка к зачёту	18	18	-
Подготовка к экзамену	27	-	27
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплны)	131	60	71
Вид промежуточной аттестации	**зачёт**	**зачёт**	**-**
	экзамен	**-**	**экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**252**	**108**	**144**

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина «Технологии разработки Web-приложений» относится к части, формируемой участниками образовательных отношений, дисциплина модуля: Технологии разработки и сопровождения информационно-коммуникационных систем, блока 1.

Изучение дисциплины базируется на результатах обучения, сформированных при изучении следующих дисциплин:

- Информатика.
- Основы программирования.
- Алгоритмы и структуры данных.
- Машинно-ориентированное программирование.

Результаты обучения по дисциплине необходимы для изучения дисциплин: «Проектирование пользовательского интерфейса», «Технологии проектирования информационных систем», «Управление качеством и тестирование ПО», «Проектный практикум» и выполнения выпускной квалификационной работы.

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Целью освоения дисциплины «Технологии разработки Web-приложений» является формирование знаний и практических навыков разработки web приложений сайтов.

Предмет дисциплины – технология разработки сценариев для Интернета.

Задачами дисциплины являются изучение принципов работы Интернета, web программирования, этапов создания web приложений.

Компетенции, формируемые в результате освоения дисциплины:

способность разрабатывать и проводить установку, настройку, оптимизацию функционирования сетевого и прикладного программного обеспечения (ПК-5);

способность проводить тестирование компонентов программного обеспечения информационных систем, осуществлять разработку, отладку, проверку работоспособности и рефакторинг программного кода (ПК-10).

В результате изучения дисциплины обучающийся должен:

Знать;

- установку, настройку, оптимизацию функционирования сетевого и прикладного программного обеспечения (ПК-5);
- тестирование компонентов программного обеспечения информационных систем, разработку, отладку, проверку работоспособности и рефакторинг программного кода (ПК-10).

Уметь:

- применять методы и средства установки, настройки, оптимизации функционирования сетевого и прикладного программного обеспечения (ПК-5);
- проводить тестирование компонентов программного обеспечения информационных систем, разработку, отладку, проверку работоспособности и рефакторинг программного кода (ПК-10).

Владеть:

- знаниями основных методов и инструментов установки, настройки, оптимизации функционирования сетевого и прикладного программного обеспечения (ПК-5);
- методами тестирования компонентов программного обеспечения информационных систем, разработки, отладки, проверки работоспособности и рефакторинг программного кода (ПК-10).

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Учебно-тематический план. Очная форма обучения. Семестр 4

Рубеж	Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем		
			Лекции	Практич. Занятия	Лабораторные работы
Рубеж 1	1	Протоколы и модели Интернет-взаимодействия	1	1	2
	2	HTML. Работа с текстом, графикой, звуком и мультимедиа	1	1	2
	3	HTML. Работа с таблицами и формами	1	1	2
	4	HTML. Работа с фреймами. Каскадные стилевые таблицы (CSS)	1	1	2
	5	Основы программирования на JavaScript	1	2	2
	6	Объектная модель документа. Динамический HTML	1	2	2
		Рубежный контроль №1	-	-	2
Рубеж 2	7	Взаимодействие с сервером и технология AJAX	2	2	2
	8	Использование СУБД MySQL в PHP	2	2	4
	9	Основы программирования на PHP	2	2	4
	10	Объектно-ориентированное программирование на PHP	2	1	4
	11	Язык XML и PHP	2	1	2
		Рубежный контроль №2	-	-	2
		Всего:	**16**	**16**	**32**

4.2. Учебно-тематический план. Очная форма обучения. Семестр 5

Рубеж	Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем	
			Лекции	Лабораторные работы
Рубеж 1	1	Серверные программы. Фреймворки	1	2
	2	Установка и настройка фреймворка Laravel	1	2
	3	Миграции фреймворка Laravel	1	2
	4	Модели фреймворка Laravel	1	2
	5	Маршрутизация фреймворка Laravel	1	2
	6	Контроллеры фреймворка Laravel	1	2
		Рубежный контроль №1	-	2
Рубеж 2	7	Шабоны фреймворка Laravel	2	4
	8	Ввод и правка данных	2	4
	9	Компоненты для клиентской части	2	4
	10	Разграничение доступа. Использование CAPTCHA	2	2
	11	Сохранение и извлечение данных	2	2
		Рубежный контроль №2	-	2
Всего:			**16**	**32**

4.3. Учебно-тематический план. Заочная форма обучения. 3 курс, зимняя сессия, 5 семестр

Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем	
		Лекции	Лабораторные работы
1	Протоколы и модели Интернет-взаимодействия	0,5	0,5
2	HTML. Работа с текстом, графикой, звуком и мультимедиа	0,5	0,5
3	HTML. Работа с таблицами и формами	0,5	0,5
4	HTML. Работа с фреймами. Каскадные стилевые таблицы (CSS)	0,5	0,5
5	Основы программирования на JavaScript	0,5	0,5
6	Объектная модель документа. Динамический HTML	0,5	0,5
7	Взаимодействие с сервером и тех-	0,5	0,5

	нология AJAX		
8	Использование СУБД MySQL в PHP	0,5	0,5
9	Основы программирования на PHP	1	1
10	Объектно-ориентированное программирование на PHP	0,5	0,5
11	Язык XML и PHP	0,5	0,5
	Всего	**6**	**6**

4.4. Учебно-тематический план. Заочная форма обучения. 3 курс, летняя сессия, 6 семестр

Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем	
		Лабораторные работы	Практические работы
1	Серверные программы. Фреймворки	0,5	0,2
2	Установка и настройка фреймворка Laravel	0,5	0,2
3	Миграции фреймворка Laravel	0,5	0,2
4	Модели фреймворка Laravel	0,5	0,2
5	Маршрутизация фреймворка Laravel	0,5	0,2
6	Контроллеры фреймворка Laravel	0,5	0,2
7	Шабоны фреймворка Laravel	0,5	0,2
8	Ввод и правка данных	0,5	1
9	Компоненты для клиентской части	1	1
10	Разграничение доступа. Использование CAPTCHA	0,5	0,4
11	Сохранение и извлечение данных	0,5	0,2
	Всего	**6**	**4**

4.5. Содержание лекционных занятий

Семестр 4

Тема 1. Протоколы и модели Интернет-взаимодействия

Интернет-соединение. Скорость передачи данных. Структура Интернета. Концептуальная модель World Wide Web (WWW). Характеристики WWW. Организация WWW. Провайдеры. Доступ в Интернет. Протоколы: Transmission Control Protocol (TCP), Internet Protocol (IP), Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP). Сетевое взаимодействие. Семи-

уровневая модель взаимодействия открытых систем (Open System Interconnection, OSI). Интернет и модель OSI. Модель клиент/сервер. Адресация в сети Интернет. Служба доменных имён DNS (Domain Name System). Порты и сетевые демоны. Структура пакетов IP и TCP. Хостинг и хостинг-провайдеры. Принципы работы общего интерфейса шлюза (Common Gateway Interface, CGI).

Тема 2.HTML. Работа с текстом, графикой, звуком и мультимедиа

Основные структурные элементы HTML-документа. Уровни заголовков. Шрифт. Абзацы и разрывы строк. Выравнивание. Форматирование. Задание начертания. Неразрывные строки. Вставка цитат. Комментарии. Акронимы. Шрифтовое выделение. Использование моноширинных шрифтов. Добавление текста с другой страницы. Расстановка пробелов. Теги.

Размещение графики на веб-странице. Графические форматы (JPEG, GIF, PNG). Пикселы и разрешение. Выравнивание изображений. Карты изображений. Масштабирование изображений. Теги.

Размещение объектов мультимедиа на веб-страницах. Анимационные файловые форматы (SWF, AVI, MOV). Форматы MPEG. Звуковые форматы (WAV, AIFF, MIDI, MP3). Потоковые данные. Технология Real Audio/Video. Технология Windows Media Server и Windows Media Player. Технология QuickTime Streaming Server и QuickTime Player.

Тема 3.HTML. Работа с таблицами и формами

Создание таблиц, строк, столбцов, заголовков и ячеек таблиц. Создание сложной табличной структуры. Группирование элементов таблиц. Атрибуты элементов таблиц. Алгоритмы обработки таблиц: фиксированный и автоматический. Теги.

Создание форм. Размещение на форме элементов управления. Списки выбора. Многострочные текстовые поля. Теги.

Тема 4. HTML. Работа с фреймами. Каскадные стилевые таблицы

Свойства фреймов. Наборы фреймов. Вставка фреймов в документ с набором (без набора) фреймов. Взаимодействие фреймов. Загрузка ссылок во фреймы. Преимущества и недостатки фреймов. Теги.

Включение CSS в HTML. Внутренние таблицы стилей. Встроенные таблицы стилей. Внешние таблицы стилей. Синтаксис CSS. Правила CSS. Типы селекторов: селектор по элементу, селектор по классу, селектор по идентификатору, контекстный селектор, псевдоэлементы, псевдоклассы, группировка селекторов. Свойства CSS. Блоки в HTML И CSS. Поля, границы, отступы, списки.

Тема 5. Основы программирования на JavaScript

Клиентские и серверные сценарии. Переменные и типы данных. Выражения. Операторы. Функции и события. Встроенные объекты JavaScript: Global, String, Number, Boolean, Array, Function, Date, Math, RegExp, Object. Свойства и методы объектов.

Тема 6. Объектная модель документа. Динамический HTML

Объектная модель браузера и документа. Родительские и дочерние объекты. Объекты браузера. Объектная модель документа (Document Object

Model, DOM). Элементы управления в формах. Создание фреймов. Создание и применение составных окон.

События динамического HTML. Всплывание событий. Связывание событий. Изменение вида элемента. События мыши. События клавиатуры. Событие прокручивания. События фокуса. События формы. События документа. Событие помощи. Объект Event. Свойства объекта Event. Динамическое содержание. Свойства динамического содержания. Объект TextRange. Свойства доступа к тексту. Метод вставки HTML. Методы позиционирования объекта TextRange. Методы управления объектом TextRange.

Тема 7. Взаимодействие с сервером и технология AJAX

Одностороннее взаимодействие с сервером: изображения, объекты, перенаправление. Двустороннее взаимодействие с сервером: изображения, файлы cookie, динамически изменяемое содержимое.

Принцип работы AJAX. Компоненты AJAX. Недостатки технологии AJAX. Создание AJAX-приложений: объект XMLHttpRequest, использование XML и создание периодических запросов, запрос данных с сервера MySQL. Библиотеки для работы с AJAX: библиотека Prototype, библиотека script.aculo.us. Библиотека EXTJS: структура библиотеки, поиск элементов (класс DomQuery), панели и компоновка элементов, формы, визуальные эффекты (Drag & drop), простые виджеты, создание редактируемых таблиц. Библиотека jQuery: функции ядра jQuery, селекторы jQuery, события в jQuery, манипуляции элементами в jQuery, AJAX-запросы в jQuery, события AJAX в jQuery, расширения для jQuery.

Тема 8. Использование СУБД MySQL в PHP

Реляционные базы данных: таблицы, записи, столбцы, отношения и ключи. Установка сервера MySQL 5 в Windows.

Запросы к базе данных: команда SELECT, запросы с указанием критерия отбора данных, группировка данных и агрегатные функции, запросы к двум и более таблицам, команды обновления и удаления данных в таблицах, изменение структуры таблицы, создание индексов, вложенные запросы

Обеспечение безопасности данных. Привелегии в MySQL. Транзакции.

Расширение *mysqli* для работы с MySQL. Клиентские и серверные части MySQL. Язык MySQL. Функции PHP для работы с MySQL: соединение PHP-сценариев с таблицами MySQL, выбор базы данных, обработка ошибок, выполнение запросов к базе данных, обработка результатов запроса, получение информации о результате.

Тема 9. Основы программирования на PHP

Редакторы для работы с PHP. Базовый синтаксис. Типы данных. Комментарии. Выражения и операторы. Константы. Переменные. Ссылки.

Операции и управляющие конструкции. Арифметические операции. Поразрядные операции. Оператор подавления ошибки. Операции сравнения. Логические операции PHP. Преобразование типов. Тернарная операция. Управляющие конструкции.

Функции и повторное использование кода. Встроенные функции. Определение и вызов пользовательских функций. Функции и область действия переменной. Статические переменные. Повторное использование кода.

Массивы. Ассоциативные массивы. Многомерные массивы. Функции для работы с массивами. Автоглобальные массивы.

Передача данных через HTML-формы. Теги формы. Работа с формами в PHP. Работа с файлами. Открытие файла. Запись в файл. Закрытие файла. Считывание данных из файла. Функции для работы с каталогами.

Строковые функции и регулярные выражения. Строки в PHP. Регулярные выражения.

Графика в PHP 5. Графические форматы данных. Подключение графической библиотеки. Создание изображений.

Тема 10. Объектно-ориентированное программирование на PHP

Объектная модель в PHP 5. Классы и объекты. Конструктор класса. Код класса и создание объекта. Деструктор объекта. Вложенные объекты. Копирование и клонирование объектов. Наследование. Финальные классы. Доступ к свойствам и методам класса. Статические свойства и методы класса. Абстрактные классы и интерфейсы. Константа класса. Ключевое слово *instance-of*. Обработка ошибок. Автозагрузка класса. Итераторы: просмотр свойств объекта.

Тема 11. Язык XML и PHP

Синтаксис XML. XML-декларация. Атрибуты. Комментарии. Процессуальная инструкция. Пространства имен XML. Особые символы. Секция CDATA (Character DATA).

Преобразование XML-документов с помощью стилевых таблиц XSL. Язык преобразования XSLT.

Применение XPath при обработке XML-документов. Выделение ветвей. Выделение нескольких путей. Выделение атрибутов. Оси и проверки узлов. Функции языка XPath.

Объектная модель документа. Объекты: *Node, NodeList, Document, Element, Attr*, Новостная лента Rich Site Summary (RSS). Создание и анализ XML-документов средствами PHP. SAX-парсер. Расширение SimpleXML в PHP 5. Расширение DOM В PHP 5: Применение DOM-функций для создания, модификации и чтения XML-документов. Расширение XSL в PHP 5.

Семестр 5

Тема 1. Серверные программы. Фреймворки

Динамические страницы и сайты. Разработка серверных программ. История веб- и PHP-фреймворков: Ruby on Rails, Codelgniter, Laravel 1,2,3,4,5. Фреймворки: модели, шаблоны, контейнеры. Философия Laravel. Преимущества Laravel. Принцип работы Laravel. Сообщество Laravel.

Тема 2. Установка и настройка фреймворка Laravel

Программные требования Laravel. Composer. Локальные среды разработки: Laravel Valet, Laravel Homestead. Создание нового проекта: установка Laravel с помощью установщика Laravel, установка Laravel с помощью функции create-project() менеджера пакетов Composer, Lambo – улучшенный ва-

риант команды laravel new. Структура папок Laravel-проекта. Настройки сайта: настройка соединения с базой данных, настройка отправки электронной почты, настройка режима работы сайта, прочие настройки.

Тема 3. Миграции фреймворка Laravel

Конфигурация: подключение базы данных, параметры конфигурации базы данных. Миграции: определение, запуск, преимущества, программирование миграций. Фасады Laravel. Создание структур данных: создание таблиц, полей, индексов, связей. Правка и переименование структур данных: добавление полей, правка и переименование полей, переименование таблиц. Удаление структур данных. Выполнение и откат миграций. Генератор запросов: использование фасада DB, чистый SQL, выстраивание цепочки с генератором запросов, транзакции. Система объектно-реляционного отображения Eloquent(Active Record ORM). Создание и определение моделей Eloquent. Получение данных с помощью Eloquent. Вставки и обновления с помощью Eloquent.Удаление с помощью Eloquent. Коллекции Eloquent.Связи в Eloquent. События Eloquent.

Тема 4. Модели фреймворка Laravel

Требования и соглашения. Создание простых моделей: прототипирование моделей, базовый класс модели, задание параметров модели. Создание связей: «один-ко-многим», «один-к-одному», «многие-ко-многим», сквозная связь. Расширение функциональности модели: создание вычисляемых полей, создание обработчиков событий, произвольные свойства и методы модели.

Тема 5. Маршрутизация фреймворка Laravel

Паттерн MVC. HTTP команды. REST – архитектурный стиль для создания API. Определения маршрутов: команды маршрутов, обработка маршрутов, параметры маршрутов, имена маршрутов. Файлы хранения настроек маршрутизации. Указание маршрутов: простые маршруты, параметризованные маршруты, правила для значений параметров в параметризованных маршрутах, именованные маршруты, указание посредников для маршрутов. Массовое создание маршрутов: базовые средства для массового создания маршрутов, дополнительные параметры массово создаваемых маршрутов. Внедрение модели в контроллер: неявное внедрение модели, явное внедрение модели. Группы маршрутов: middleware, префиксы путей, запасные маршруты, префиксы пространства имён, префиксы имён. Подписанные маршруты: подписание маршрута, изменение маршрутов для разрешения подписанных ссылок, Физические интернет-адреса.

Тема 6. Контроллеры фреймворка Laravel

Требования и соглашения. Создание контроллеров. Получение данных от посетителя. Работа с базой данных. Простая выборка записей: поиск записей по номерам, выборка всех записей, выборка первой записи. Получение значений полей записи. Получение связанных записей. Создание запросов к базе данных: фильтрация записей, фильтрация по наличию или отсутствию связанных записей, сортировка записей, указание выбираемых полей, выборка уникальных записей, связывание таблиц. Использование агрегатных функций. Группировка записей: получение количества связанных записей,

использование агрегатных функций применительно ко всем записям, использование агрегатных функций применительно к сгруппированным записям. Фильтрации и сортировка групп записей. Ограничение количества выбираемых записей. Специальные случаи выборки записей. Использование пагинатора: упрощенный пагинатор, полнофункциональный пагинатор, Получение сведений о запросе. Получение путей к папкам фреймворка. Вывод данных. Перенаправление. Указание посредников в контроллерах. Особые разновидности контроллеров: контроллеры-функции, контроллеры-действия.

Тема 7. Шаблоны фреймворка Laravel

Требования и соглашения. Создание шаблонов. Язык шаблонов Blade. Отображение данных. Управляющие структуры: условные конструкции, циклы. Наследование шаблонов: определение разделов страницы с помощью директив @section, @show, @yield. Включение составляющих представления: использование стеков, использование компонентов и слотов. Компоновщики представлений и внедрение сервисов: привязка данных к представлениям с использованием компоновщиков представлений, внедрение сервиса Blade. Пользовательские директивы Blade: параметры пользовательских директив Blade,упрощённые пользовательские директивы для операторов if. Комментарии Blade.Вставка PHP-кода. Особые случаи вывода данных: генерирование интернет-адресов, создание web-форм и элементов управления, вывод всплывающих сообщений, вывод пагинатора. Вложенные шаблоны. Наследование шаблонов: создание шаблонов-родителей, создание шаблонов-потомков. Стеки. Разделяемые данные и составители. Получение доступа к контроллеру.

Тема 8. Ввод и правка данных

Создание, правка и удаление отдельных записей: создание web-формы для ввода и правки записи, создание записи, правка записи, удаление записи, обработка связей между таблицами. Дополнительные инструменты для создания и правки отдельных записей: поиск или создание записей, исправление или создание записей, работа со связанными записями. Проверка введённых в форму данных на корректность. Валидаторы. Простейшие валидаторы: автоматическая валидация, полуавтоматическая валидация, условные правила ввалидации. Правила валидации. Запросы форм. Массовое создание, правка и удаление записей. Работа с выгруженными файлами: файловое хранилище и диски laravel, создание web-формы для выгрузки файлов, получение и сохранение выгруженных файлов, работа с выгруженными файлами.

Тема 9. Компоненты для клиентской части

Laravel Mix. Структура каталога Mix. Запуск Mix. Предустановки клиентской части и генерация кода аутентификации. Разбивка на страницы: разбивка на страницы результатов из базы данных, создание разбивщиков страниц вручную. Панель сообщений. Строковые хелперы, множественность и локализация: строковые хелперы и множественность, локализация. Тестирование: тестирование пакетов сообщений и ошибок, перевод и локализация.

Тема 10. Разграничение доступа. Использование CAPTCHA

Встроенная система разграничения доступа Laravel. Ограничение доступа к страницам: простые случаи ограничения доступа, ограничение доступа на основе сложных условий (шлюзы, политики), ограничение доступа на основе запросов форм, ограничение доступа в шаблонах. Вывод сведений о текущем пользователе. Настройка встроенной подсистемы разграничения доступа: базовые настройки, модификация списка пользователей, настройка писем, отправляемых действием восстановления пароля (создание оповещения, подготовка шаблона для оповещения, отправка оповещений). Использование CAPTCHA. Библиотека Captcha for Laravel: установка, настройка, использование.

Тема 11. Сохранение и извлечение данных

Локальные и облачные файловые менеджеры: настройка доступа к файлам, фасад Storage, пакет Flysystem. Базове способы загрузки файлов на сервер и манипулирование файлами. Простые способы скачивания файлов. Сессии: получение доступа к сессии, методы, доступные в экземплярах сессий, флеш-память сессии. Настройки кэширования: кэширование в папке, кэширование в таблице базы данных, кэширование в оперативной памяти. Занесение данных в кэш. Изменение данных, хранящихся в кэше. Получение данных из кэша: простое получение данных, получение данных из кэша с одновременной их записью, получение данных из кэша с последующим удалением. Удаление данных из кэша. Cookie-файлы: cookie-файлы в Laravel, получение доступа к cookie-файлам. Логирование: внесение записей в логи, каналы логирования. Полнотекстовый поиск с использованием Laravel Scout: пакет Scout, пометка модели для индексирования, поиск по индексу, очереди и Scout, выполнение операций без индексирования, условное индексирование моделей, запуск индексирования вручную с помощью кода, запуск индексирования вручную с помощью интерфейса командной строки. Работа с другим хранилищем.

4.6. Лабораторные занятия. Очная форма обучения. Семестр 4

Номер раздела, темы	Наименование раздела, Темы	Наименование лабораторной работы	Норматив времени, час.
1	Протоколы и модели Интернет-взаимодействия	Разработка сайта, содержащего изображения и гиперссылки на web-страницы.	2
2	HTML. Работа с текстом, графикой, звуком и мультимедиа	Разработка сайта, содержащего карту-изображение с заданными активными зонами.	2
3	HTML. Работа с таблицами и формами	Разработка сайта, содержащего таблицы.	2

4	HTML. Работа с фреймами. Каскадные стилевые таблицы (CSS)	Разработка сайта «Электронный тест проверки знаний студентов изучаемых дисциплин».	2
5	Основы программирования на JavaScript	Разработка сайта с динамическими эффектами.	2
6	Объектная модель документа. Динамический HTML	JavaScript. Горизонтальное и вертикальное меню.	2
		Рубежный контроль №1	2
7	Взаимодействие с сервером и технология AJAX	Технология фонового обмена данными браузера с веб-сервером в информационных сетях	2
8	Использование СУБД MySQL в PHP	Динамическая поддержка выбора значений из предлагаемого списка	4
9	Основы программирования на PHP	Проектирование сайта электронной коммерции	4
10	Объектно-ориентированное программирование на PHP	Разработка Интернет магазина	4
11	Язык XML и PHP	Разработка SAX-парсера новостной ленты	2
		Рубежный контроль №2	2
		Всего:	**32**

4.7. Лабораторные занятия. Очная форма обучения. Семестр 5

Номер раздела, темы	**Наименование раздела, Темы**	**Наименование лабораторной работы**	**Норматив времени, час.**
1	Серверные программы. Фреймворки	Установка и настройка WAMP-Open Server Panel	2
2	Установка и настройка фреймворка Laravel	Настройка среды разработки для использования Laravel	2
3	Миграции фреймворка Laravel	Миграции. Создание миграций	2
4	Модели фреймворка Laravel	Модели. Создание моделей.	2
5	Маршрутизация фреймворка Laravel	Маршрутизация. Создание маршрутов	2
6	Контроллеры фреймворка Laravel	Контроллеры. Создание контроллеров	2
		Рубежный контроль №1	2
7	Шабоны фреймворка Laravel	Шаблоны. Создание шаблонов	4
8	Ввод и правка данных	Работа с формами. Валидация в Laravel	4
9	Компоненты для клиентской части	Создание разбивщиков страниц вручную	4

10	Разграничение доступа. Использование CAPTCHA	Разграничение доступа и использование CAPTCHA	2
11	Кэширование	Кэширование данных	2
		Рубежный контроль №2	2
		Всего:	**32**

4.8. Практические занятия. Очная форма обучения. Семестр 4

Номер раздела, темы	**Наименование раздела, Темы**	**Наименование практической работы**	**Норматив времени, час.**
1	Протоколы и модели Интернет-взаимодействия	Разработка сайта, содержащего изображения и гиперссылки на web-страницы.	1
2	HTML. Работа с текстом, графикой, звуком и мультимедиа	Разработка сайта, содержащего карту-изображение с заданными активными зонами.	1
3	HTML. Работа с таблицами и формами	Разработка сайта, содержащего таблицы.	1
4	HTML. Работа с фреймами. Каскадные стилевые таблицы (CSS)	Разработка сайта «Электронный тест проверки знаний студентов изучаемых дисциплин».	1
5	Основы программирования на JavaScript	Разработка сайта с динамическими эффектами.	2
6	Объектная модель документа. Динамический HTML	JavaScript. Горизонтальное и вертикальное меню.	2
7	Взаимодействие с сервером и технология AJAX	Технология фонового обмена данными браузера с веб-сервером в информационных сетях	2
8	Использование СУБД MySQL в PHP	Динамическая поддержка выбора значений из предлагаемого списка	2
9	Основы программирования на PHP	Проектирование сайта электронной коммерции	2
10	Объектно-ориентированное программирование на PHP	Разработка Интернет магазина	1
11	Язык XML и PHP	Разработка SAX-парсера новостной ленты	1
		Всего:	**16**

4.9. Лабораторные занятия. Заочная форма обучения. 3 курс, зимняя Сессия, 5 семестр

Номер раздела, темы	Наименование раздела, Темы	Наименование лабораторной Работы	Норматив времени, час.
1	Протоколы и модели Интернет-взаимодействия	Разработка сайта, содержащего изображения и гиперссылки на web-страницы.	0,5
2	HTML. Работа с текстом, графикой, звуком и мультимедиа	Разработка сайта, содержащего карту-изображение с заданными активными зонами.	0,5
3	HTML. Работа с таблицами и формами	Разработка сайта, содержащего таблицы.	0,5
4	HTML. Работа с фреймами. Каскадные стилевые таблицы (CSS)	Разработка сайта «Электронный тест проверки знаний студентов изучаемых дисциплин».	0,5
5	Основы программирования на JavaScript	Разработка сайта с динамическими эффектами.	0,5
6	Объектная модель документа. Динамический HTML	JavaScript. Горизонтальное и вертикальное меню.	0,5
7	Взаимодействие с сервером и технология AJAX	Технология фонового обмена данными браузера с веб-сервером в информационных сетях	0,5
8	Использование СУБД MySQL в PHP	Динамическая поддержка выбора значений из предлагаемого списка	0,5
9	Основы программирования на PHP	Проектирование сайта электронной коммерции	1
10	Объектно-ориентированное программирование на PHP	Разработка Интернет магазина	0,5
11	Язык XML и PHP	Разработка SAX-парсера новостной ленты	0,5
		Всего:	**6**

4.10. Лабораторные занятия. Заочная форма обучения. 3 курс, летняя сессия, 6 семестр

Номер раздела, темы	Наименование раздела, Темы	Наименование лабораторные Работы	Норматив времени, час.
1	Серверные программы. Фреймворки	Установка и настройка WAMP-Open Server Panel	0,5
2	Установка и настройка фреймворка Laravel	Настройка среды разработки для использования Laravel	0,5
3	Миграции фреймворка Laravel	Миграции. Создание миграций	0,5
4	Модели фреймворка Laravel	Модели. Создание моделей.	0,5
5	Маршрутизация фреймворка Laravel	Маршрутизация. Создание маршрутов	0,5
6	Контроллеры фреймворка Laravel	Контроллеры. Создание контроллеров	0,5
7	Шабоны фреймворка Laravel	Шаблоны. Создание шаблонов	0,5
8	Ввод и правка данных	Работа с формами. Валидация в Laravel	0,5
9	Компоненты для клиентской части	Создание разбивщиков страниц вручную	1
10	Разграничение доступа. Использование CAPTCHA	Разграничение доступа и использование CAPTCHA	0,5
11	Кэширование	Кэширование данных	0,5
		Всего:	**6**

4.11. Практические занятия. Заочная форма обучения. 3 курс, летняя сессия, 6 семестр

Номер раздела, темы	Наименование раздела, Темы	Наименование практической работы	Норматив времени, час.
1	Серверные программы. Фреймворки	Установка и настройка WAMP-Open Server Panel	0,2
2	Установка и настройка фреймворка Laravel	Настройка среды разработки для использования Laravel	0,2
3	Миграции фреймворка Laravel	Миграции. Создание миграций	0,2
4	Модели фреймворка Laravel	Модели. Создание моделей.	0,2

5	Маршрутизация фреймворка Laravel	Маршрутизация. Создание маршрутов	0,2
6	Контроллеры фреймворка Laravel	Контроллеры. Создание контроллеров	0,2
7	Шабоны фреймворка Laravel	Шаблоны. Создание шаблонов	0,2
8	Ввод и правка данных	Работа с формами. Валидация в Laravel	1
9	Компоненты для клиентской части	Создание разбивщиков страниц вручную	1
10	Разграничение доступа. Использование CAPTCHA	Разграничение доступа и использование CAPTCHA	0,4
11	Кэширование	Кэширование данных	0,2
		Всего:	**4**

4.12. Контрольная работа

Контрольная работа посвящена разработке сайта, по вариантам задания, согласно методических рекомендаций, указанных в разделе 8.

4.13. Курсовой проект

Курсовой проект посвящен разработке сайта, по вариантам задания, согласно методических рекомендаций, указанных в разделе 8.

5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Лекционный курс основывается на методе обучения, использующем технологию, при которой студенты конспектируют теоретический материал, участвуют в опросах и дискуссиях. В этом случае задействованы зрительная, слуховая, моторная и ассоциативная виды памяти.

При прослушивании лекций рекомендуется в конспекте отмечать все важные моменты, на которых заостряет внимание преподаватель, в частности те, которые направлены на качественное выполнение соответствующей лабораторной и практической работ.

Преподавателем запланировано использование при чтении лекций технологии учебной дискуссии. Поэтому рекомендуется фиксировать для себя интересные моменты с целью их активного обсуждения на дискуссии в конце лекции.

Залогом качественного выполнения лабораторных и практических работ является самостоятельная подготовка к ним накануне путем повторения материалов лекций. Рекомендуется подготовить вопросы по неясным моментам и обсудить их с преподавателем в начале занятия.

Лабораторные и практические работы выполняются с применением HTML, CSS, JavaScript, PHP 5, СУБД MySQL, Laravel и новых версий этих программных продуктов.

Преподавателем запланировано применение на лабораторных и практических занятиях технологий развивающейся кооперации, коллективного взаимодействия, разбора конкретных ситуаций.

Для текущего контроля успеваемости по очной форме обучения преподавателем используется балльно-рейтинговая система контроля и оценки академической активности. Поэтому настоятельно рекомендуется тщательно прорабатывать материал дисциплины при самостоятельной работе, участвовать во всех формах обсуждения и взаимодействия, как на лекциях, так и на лабораторных и практических занятиях в целях лучшего освоения материала и получения высокой оценки по результатам освоения дисциплины.

Выполнение самостоятельной работы подразумевает самостоятельное изучение разделов дисциплины, подготовку к лабораторным и практическим занятиям, к рубежным контролям (для очной формы обучения), выполнение контрольной работы и курсового проекта, подготовку к зачёту и экзамену.

Рекомендуемая трудоемкость самостоятельной работы для очной формы обучения представлена в таблице:

Рекомендуемый режим самостоятельной работы
Очная форма

Наименование вида самостоятельной работы	**Рекомендуемая трудоемкость, акад. час.**	
	4 семестр	**5 семестр**
Самостоятельное изучение тем дисциплины:	**-**	**22**
Контейнер	-	5
Аутентификация и авторизация пользователей	-	5
Тестирование команд Artisan и систем Laravel	-	6
Экосистема инструментов Laravel	-	6
Подготовка к лабораторным занятиям (по 0,5 часа на каждое занятие)	**-**	**7**
Подготовка к практическим занятиям (по 0,5 часа на каждое занятие)	**4**	**-**
Подготовка к рубежным контролям (по 2 часа на каждый рубеж)	**4**	**4**
Выполнение контрольной работы	**18**	**-**
Выполнение курсового проекта	**-**	**36**
Подготовка к зачёту	**18**	**-**
Подготовка к экзамену	**-**	**27**
Всего:	**44**	**96**

Рекомендуемая трудоемкость самостоятельной работы для заочной формы обучения представлена в таблице:

Рекомендуемый режим самостоятельной работы
Заочная форма

Наименование вида самостоятельной работы	**Рекомендуемая трудоемкость, акад. час.**	
	3 курс	
	Зимняя сессия	**Летняя сессия**
Самостоятельное изучение тем дисциплины:	**57**	**66**
Контейнер	14	16
Аутентификация и авторизация пользователей	14	16
Тестирование команд Artisan и систем Laravel	14	17
Экосистема инструментов Laravel	15	17
Подготовка к лабораторным занятиям (по 1 часу на каждое занятие)	**3**	**3**
Подготовка к практическим занятиям (по 1 часу на каждое занятие)	**-**	**2**
Выполнение контрольной работы	**18**	**-**
Выполнение курсового проекта	**-**	**36**
Подготовка к зачёту	**18**	**-**
Подготовка к экзамену	**-**	**27**
Всего:	**96**	**134**

6. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

6.1. Перечень оценочных средств

1. Балльно-рейтинговая система контроля и оценки академической активности студентов в КГУ (для очной формы обучения).
2. Отчеты студентов по лабораторным работам.
3. Отчеты студентов по практическим занятиям.
4. Банк заданий к рубежным контролям № 1, № 2 (для очной формы обучения).
5. Банк заданий к зачёту.
6 Банк заданий к экзамену.
7 Контрольная работа.
6. Курсовой проект.

6.2. Система балльно-рейтинговой оценки работы студентов по дисциплине

Очная форма обучения

<table>
<tr><td>№</td><td>Наименование</td><td colspan="8">Содержание</td></tr>
<tr><td rowspan="4">1</td><td rowspan="4">Распределение баллов за семестры по видам учебной работы, сроки сдачи учебной работы (доводятся до сведения студентов на первом учебном занятии)</td><td colspan="8">Распределение баллов,
4 семестр</td></tr>
<tr><td>Вид учебной работы:</td><td>Посещение лекций</td><td>Выполнение и защита отчетов по лабораторным работам</td><td>Посещение практических занятий</td><td>Рубежный контроль №1</td><td>Рубежный контроль №2</td><td>Контрольная работа</td><td>Зачёт</td></tr>
<tr><td>Балльная оценка:</td><td>2б*8=16б</td><td>3б*11=33б</td><td>1б*8=8б</td><td>5</td><td>5</td><td>3</td><td>30</td></tr>
<tr><td colspan="8">Распределение баллов,
5 семестр</td></tr>
<tr><td></td><td>Вид учебной работы:</td><td>Посещение лекций</td><td colspan="2">Выполнение и защита отчетов по лабораторным работам</td><td>Рубежный контроль №1</td><td>Рубежный контроль №2</td><td>Экзамен</td></tr>
<tr><td></td><td>Балльная оценка:</td><td>2б*8=16б</td><td colspan="2">4б*11=44б</td><td>5</td><td>5</td><td>30</td></tr>
<tr><td>2</td><td colspan="2">Критерий пересчета баллов в традиционную оценку по итогам работы в семестре и зачёта</td><td colspan="7">60 и менее баллов – незачтено;
61…73 – зачтено;
74… 90 – зачтено;
91…100 – зачтено</td></tr>
</table>

3	Критерии допуска к промежуточной аттестации, возможности получения автоматического зачета (экзаменационной оценки) по дисциплине, возможность получения бонусных баллов	Для допуска к промежуточной аттестации по дисциплине за семестр обучающийся должен набрать по итогам текущего и рубежного контролей не менее 51 балла. В случае если обучающийся набрал менее 51 балла, то к аттестационным испытаниям он не допускается. Для получения экзамена или зачета без проведения процедуры промежуточной аттестации обучающемуся необходимо набрать в ходе текущего и рубежных контролей не менее 61 балла. В этом случае итог балльной оценки, получаемой обучающимся, определяется по количеству баллов, набранных им в ходе текущего и рубежных контролей. При этом, на усмотрение преподавателя, балльная оценка обучающегося может быть повышена за счет получения дополнительных баллов за академическую активность. Обучающийся, имеющий право на получение оценки без проведения процедуры промежуточной аттестации, может повысить ее путем сдачи аттестационного испытания. В случае получения обучающимся на аттестационном испытании 0 баллов итог балльной оценки по дисциплине не снижается. За академическую активность в ходе освоения дисциплины, участие в учебной, научно-исследовательской, спортивной, культурно-творческой и общественной деятельности обучающемуся могут быть начислены дополнительные баллы. Максимальное количество дополнительных баллов за академическую активность составляет 30. Основанием для получения дополнительных баллов являются: - выполнение дополнительных заданий по дисциплине; дополнительные баллы начисляются преподавателем; - участие в течение семестра в учебной, научно-исследовательской, спортивной, культурно-творческой и общественной деятельности КГУ.
4	Формы и виды учебной работы для неуспевающих (восстановившихся на курсе обучения) студентов для получения недостающих баллов в конце семестра	В случае если к промежуточной аттестации (экзамену, зачету) набрана сумма менее 51 балла, обучающемуся необходимо набрать недостающее количество баллов за счет выполнения дополнительных заданий, до конца последней (зачетной) недели семестра. Ликвидация академических задолженностей, возникших из-за разности в учебных планах при переводе или восстановлении, проводится путем выполнения дополнительных заданий, форма и объем которых определяется преподавателем.

5	Критерии оценки курсового проекта	По курсовому проекту выставляется отдельная оценка. Максимальная сумма по курсовому проекту устанавливается в 100 баллов. При оценке качества выполнения курсового проекта и уровня защиты рекомендуется следующее распределение баллов: а) качество пояснительной записки и графической части – до 40 баллов; б) качество доклада – до 20 баллов; в) качество защиты проекта – до 40 баллов. При рассмотрении качества пояснительной записки и графической части курсового проекта принимается к сведению ритмичность выполнения проекта, отсутствие ошибок, логичность и последовательность построения материала, правильность выполнения и полнота расчетов, соблюдение требований к оформлению и аккуратность исполнения курсового проекта. При оценке качества доклада учитывается уровень владения материалом, степень аргументированности, четкости, последовательности и правильности изложения материала, а также соблюдение регламентов. При оценке уровня качества ответов на вопросы принимается во внимание правильность, полнота и степень ориентированности в материале. Комиссия по приему защиты курсового проекта оценивает вышеуказанные составляющие компоненты и определяет итоговую оценку.

6.3. Процедура оценивания результатов освоения дисциплины

Рубежные контроли, зачёт и экзамен проводятся в виде ответов на вопросы.

Перед проведением рубежного контроля преподаватель прорабатывает со студентами основной материал соответствующих разделов дисциплины в форме краткой лекции-дискуссии.

На подготовку при рубежном контроле студенту отводится 1 академический час.

Варианты заданий для рубежных контролей № 1, № 2 состоят из 20 вопросов. Для определения баллов при проверке рубежных контролей используются интервальные оценки, представленные в таблице

Количество правильных ответов	1-5	6-8	9-11	12-14	15-17	18-20
Количество баллов	0	1	2	3	4	5

Преподаватель оценивает в баллах результаты рубежного контроля каждого студента по количеству правильных ответов и заносит в ведомость учета текущей успеваемости.

Зачёт состоит из 1 вопроса. Вопросы к зачёту доводятся до студентов на последней лекции в семестре. Вопрос оценивается в 30 баллов. На подготовку ответа студенту отводится 1 астрономический час.

Результаты текущего контроля успеваемости и зачёта заносятся преподавателем в зачётную ведомость, которая сдается в организационный отдел института на зачётной неделе, а также выставляются в зачетную книжку студента.

Экзамен состоит из 2 вопросов. Вопросы к экзамену доводятся до студентов на последней лекции в семестре. Каждый вопрос оценивается в 15 баллов. На подготовку ответа студенту отводится 1 астрономический час.

Результаты текущего контроля успеваемости и экзамена заносятся преподавателем в экзаменационную ведомость, которая сдается в организационный отдел института в день экзамена, а также выставляются в зачетную книжку студента.

6.4. Примеры оценочных средств для рубежных контролей, зачёта и экзамена

6.4.1 Примеры заданий для рубежного контроля №1
Очная форма обучения, 4 семестр

ВАРИАНТ 1_1

1. Кто является создателем языка HTML?

1. Тим Бернерс-Ли
2. Сергей Брин.
3. Рик Масситт.
4. Sun Microsystems.

2. В каком году был создан HTML?

1. 2000
2. 1988.
3. 1917.
4. 1989.
5. 2002.

3. Какой протокол использует Web-сервер для передачи файлов по Internet?

1. HTML;
2. HTPP;
3. FTP;
4. IP.

4. В каком порядке следует разместить основные дескрипторы, составляющие шаблон HTML-документа?

1. <html><head><title></title></head><body></body></html>
2. <html></html><head></head><title></title><body></body>
3. <html></html><head><body></body></head><title></title>
4. <html><head></head><title></title><body></body><html>

5. Для чего используется атрибут target тега <a>?(HTML)

1. задает адрес документа, по которому следует перейти.
2. устанавливает имя якоря внутри документа.
3. задает имя окна или фрейма, куда браузер будет загружать документ
4. добавляет всплывающую подсказку к тексту ссылки.
5. атрибут target недопустим для тега <a>.

6. Какой html-тег используется для создания заголовков наибольшего размера?

1. <heading>.
2. <head>.
3. <large>.
4. <h1>.
5. <head>.

7. В выражении <body bgcolor='blue'> bgcolor – это

1. Элемент.
2. Стиль.
3. Атрибут.
4. Дескриптор.

8. В чём заключается назначение CSS?

1. В создании интерактивных сайтов.
2. В разделении содержания и представления веб-страницы
3. В структуризации контента.
4. В создании большой таблицы.

9. Как правильно обозначается селектор идентификатора?(CSS)

1. .id1.
2. @id1.
3. ~id1.
4. #id1.
5. -idl.

10. Назовите свойство CSS для задания размера шрифта:

1. size-font.
2. font-size.
3. font-family.
4. font-font.

11. Какие свойства в CSS регулируют расположение обтекаемых блоков?(CSS)

1. float
2. center.
3. clear.
4. relative.
5. go to.

12. Как правильно обозначается селектор класса?(CSS)

1. .class1
2. @class1.
3. #class1.

4. ~class1.
5. ~--class1.

13. Какое свойство CSS задаёт расстояние от содержимого элемента до рамки:

1. left.
2. margin.
3. padding.
4. top.
5. aligan.

14. Какое значение не может принять свойство display?(CSS)

1. block.
2. position.
3. none.
4. inline.
5. shoot.

15. Какое значение не может принимать свойство text-align?(CSS)

1. large.
2. center.
3. left.
4. justify.
5. right.

16. Какое значение не может принять свойство list-style-type?(CSS)

1. decimal.
2. upper-roman.
3. small.
4. upper-roman.

17. Укажите возможный в CSS тип селекторов

1. Селектор по возможности.
2. Селектор по маске.
3. Селектор по странице.
4. Селектор по классу.

18. Какое значение в параметре background-repeat задаст повторение изображения по горизонтали:(CSS)

1. repeat-y.
2. repeat-x.
3. repeat.
4. no-repeat.
5. repeat-z.

19. Какое значение в параметре background-color позволяет сделать фон прозрачным:(CSS)

1. transparent.
2. inherit.
3. justify.
4. repeat.
5. right.

20. Какой параметр может задать до 5 свойств фона:(CSS)

1. background-repeat.
2. background-attachment.
3. background.
4. background-image.
5. background-color.
6. color.
7. background-position.

ВАРИАНТ 1_2

1. Управление какими объектами может осуществляться при помощи JavaScript?

1. Java-апплетами;
2. роликами Flash;
3. объектами Flash;
4. анимационными картинками.

2. Какие атрибуты определяют язык сценария?

1. LANGUAGE;
2. SCRIPT;
3. SRC;
4. TYPE;
5. NOSCRIPT.

3. Что выведет на экран следующая строка сценария var a=null; document.write (++a)?

1. null;
2. 1;
3. null1;
4. NaN.

4. Для каких значений переменной value условие value%5==0 даст значение true?

1. 0;
2. 1;
3. 50;
4. 6.

5. Какие из перечисленных ключевых слов служат для создания цикла?

1. for;
2. break;
3. continue;
4. while;
5. switch.

6. Что будет выведено в окне браузера в результате выполнения инструкции document.write(eval('document.write("Привет")'))?

1. undefined;

2. Привет;
3. undefined Привет;
4. Привет undefined.

7. Какие из перечисленных выражений не являются корректным обращением к свойству объекта?

1. myHouse.window;
2. myHouse['windows'][1];
3. myHouse.'window';
4. myHouse.windows.1.

8. Какие из перечисленных инструкций не могут добавить новых свойств объекту myHouse?

1. myHouse.heigth=7;
2. House.prototype.heigth=null;
3. document.write (myHouse.height);
4. document.write (myHouse.setHeigth(7)).

9. Какие из перечисленных объектов являются дочерними по отношению к объекту window?

1. document;
2. History;
3. screen;
4. Date.

10. Какие из перечисленных выражений могут быть указателями на объект элемента <IMG ID="logo"> SRC="logo.jpg"?

1. document.images.logo;
2. document.images [0];
3. document.getElementById('LOGO');
4. document.getElementByTagName('logo').

11. Какой из атрибутов тега FRAME определяет указатель к соответствующему объекту window?

1. SRC;
2. NAME;
3. ID;
4. HREF.

12. Какой из перечисленных шаблонов регулярных выражений соответствует любому слову, записанному с прописной буквы?

1. /[а-я]+/;
2. /[А-Я][а-я]*/;
3. /[А-Яа-я]+/;
4. /[А-Я]+[а-я]+/.

13. Какой из перечисленных обработчиков событий чаще всего применяется для кнопок?

1. ONCHANGE;
2. ONCLICK;
3. ONKEYPRESS;
4. ONSUBMIT;

5. ONMOUSEMOVE.

14. Как нужно задать стиль форматирования элемента, чтобы его настройки можно было прочитать с помощью свойства style соответствующего объекта?

1. тегом STYLE;
2. с помощью атрибута STYLE;
3. тегом LINK;
4. посредством сценария JavaScript;
5. с помощью правила @import.

15. Посредством какой из функций, входящей в состав библиотеки LayerLibrary.js можно начать динамическое построение нового слоя в документе?

1. getLStyle();
2. makeLayer();
3. finLayer();
4. setLBg().

16. Для вставки Java-аплетов в HTML-документ может использоваться тег

1. <JAVA>;
2. <APPLET>;
3. <JAVA_APPLET>;
4. <OBJECT>;
5. <JAVA_OBJECT>.

17. Элементы ActiveX отличаются от Java-аплетов тем, что

1. имеют более широкие возможности доступа к системе пользователя;
2. являются более универсальными и поддерживаются любыми браузерами на любых платформах;
3. представляют собой выполняемый программный код;
4. ничем не отличаются – это лишь другое название одной и той же технологии.

18. Какие из форм коммуникаций можно использовать при передаче информации от браузера к серверу?

1. односторонняя коммуникация;
2. двухсторонняя коммуникация;
3. трехсторонняя коммуникация;
4. многосторонняя коммуникация.

19. Для организации простейшего обмена данными сценария на языке JavaScript с сервером можно использовать HTML-теги

1. <A>;
2. <IMG>;
3. <DIV>;
4. <H1>;
5. теоретически любой тег, имеющий атрибут SRC.

20. Что из перечисленного ниже не является компонентом технологии AJAX

1. ActiveX;
2. CSS;
3. DOM;
4. JavaScript;
5. XMLHttpRequest.

6.4.2 Примеры заданий для рубежного контроля №2
Очная форма обучения, 4 семестр

ВАРИАНТ 2_1

1. Универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных?

1. SQL
2. PHP.
3. Delphi.
4. HTML.

2. Какая из этих строковых функций SQL допустима?

1. BINARY().
2. OUTER().
3. UPPER()
4. CHOP().
5. SPLIT().

3. Что делает утилита командной строки Isamchk?

1. Читает создаваемые MySQL журналы, относящиеся к ISAM-файлам.
2. Производит проверку файлов, содержащих данные базы
3. Модифицирует таблицы прав доступа MySQL и отображает их в удобном для чтения виде.
4. Считывает данные из файла и вводит их в таблицу базы данных.

4. Что делает утилита командной строки Mysqlshow?

1. Читает создаваемые MySQL журналы, относящиеся к ISAM-файлам.
2. Выводит на экран структуру баз данных, имеющихся на сервере, и таблицы, из которых они состоят
3. Модифицирует таблицы прав доступа MySQL и отображает их в удобном для чтения виде.
4. Считывает данные из файла и вводит их в таблицу базы данных.

5. Что делает утилита командной строки Mysqlimport ?

1. Читает создаваемые MySQL журналы, относящиеся к ISAM-файлам.
2. Производит проверку файлов, содержащих данные базы.
3. Модифицирует таблицы прав доступа MySQL и отображает их в удобном для чтения виде.
4. Считывает данные из файла и вводит их в таблицу базы данных

6. Что делает утилита командной строки Isamlog ?

1. Читает создаваемые MySQL журналы, относящиеся к ISAM-файлам

2. Производит проверку файлов, содержащих данные базы.

3. Модифицирует таблицы прав доступа MySQL и отображает их в удобном для чтения виде.

4. Считывает данные из файла и вводит их в таблицу базы данных.

7. При каком соединении могут остаться висящие кортежи?

1. при тета-соединении (theta-join).
2. при перекрестном соединении (cross join).
3. при левом внешнем соединении (left outer join)
4. при естественном соединении (natural join).

8. Какова максимальная длина строки типа TEXT(150)?

1. 255.
2. 65535
3. 16777215.
4. 4294967295.

9. Какова максимальная длина строки типа VARCHAR(150)?

1. 255
2. 65535.
3. 16777215.
4. 4294967295.

10. Для чего предназначена агрегатная функция MySQL BIT_OR(expression)?

1. Возвращает результат побитового И, агрегирующего все значения в expression.
2. Возвращает среднеквадратичное отклонение значения в expression.
3. Возвращает побитовое ИЛИ, агрегирующее все значения в expression
4. Возвращает среднее значение из значений в expression.

11. Для чего предназначена агрегатная функция MySQL BIT_AND(expression)?

1. Возвращает результат побитового И, агрегирующего все значения в expression
2. Возвращает среднеквадратичное отклонение значения в expression.
3. Возвращает побитовое ИЛИ, агрегирующее все значения в expression.
4. Возвращает среднее значение из значений в expression.

12. Для чего предназначена агрегатная функция MySQL STD(expression)?

1. Возвращает результат побитового И, агрегирующего все значения в expression.
2. Возвращает среднеквадратичное отклонение значения в expression
3. Возвращает побитовое ИЛИ, агрегирующее все значения в expression.
4. Возвращает среднее значение из значений в expression.

13. Для чего применяются индексы в БД?

1. для успешного завершения транзакций.
2. для объединения таблиц.

3. для отката изменений.

4. для ускорения доступа к данным

14. Какой из вариантов не является функцией СУБД?

1. обеспечение пользователя языковыми средствами манипулирования данными.

2. координация проектирования, реализации и ведения БД

3. поддержка моделей пользователя.

4. реализация языков определения и манипулирования данными.

5. защита и целостность данных.

15. Какого типа данных нет в MySQL?

1. date.

2. string

3. enum.

4. char.

16. Какой из перечисленных типов подойдет для хранения даты:

1. BIT.

2. DATETIME

3. YEAR.

4. TIME.

17. Является ли тип SET символьным типом?

1. Да

2. Нет.

18. Какой оператор используется для получения информации о таблице?

1. INFO.

2. SHOW.

3.DESCRIBE

4. INFORMATION.

5. SELECT.

19. В чем различия между выражениями UNION ALL и UNION?

1. UNION быстрее выполнится, чем UNION ALL.

2. При применении UNION ALL результат запроса не будет содержать повторяющиеся строки.

3. При применении UNION результат запроса не будет содержать повторяющиеся строки

20. С помощью какой конструкции можно посмотреть, сколько баз данных есть в системе?

1. check databases.

2. select count(*) from databases.

3. view databases.

4. count databases.

5. show databases

ВАРИАНТ 2_2

1. Кто является создателем языка PHP?

1. Расмус Лердорф
2. Сергей Брин.
3. Рик Масситт.
4. Sun Microsystems.

2. В каком году был создан язык PHP?

1. 1994
2. 1988.
3. 1917.
4. 2000.

3. Что означает аббревиатура PHP?

1. Personal Home Page
2. PHP Hypertext Preprocessor.
3. ASP Server Pages.
4. XML Markup Language.

4. Какая функция PHP возвращает содержимое всех полей одной записи из всего набора записей, возвратившихся в результате запроса к базе данных сервера MySQL, в виде индексированного массива?

1. mysql_field().
2. mysql_result().
3. mysql_records().
4. mysql_fetch_array()

5. Что из перечисленного является предопределенными константами PHP?

1. __LINE__
2. E_WARNING.
3. PHPINFO.
4. $E_ALL.

6. Какая функция PHP возвращает имя сессии?

1. session_register().
2. session_name()
3. session_id().
4. session().

7. Какие функции PHP используются для определения типа переменных?

1. is_int()
2. settype().
3. is_double().
4. object().

8. Что возвращает функция mysql_connect() в случае неудачного соединения с MySQL сервером?

1. номер ошибки подключения.
2. дескриптор подключения к серверу.

3. причину неудачного подключения.
4. логическое значение FALSE

9. Какой оператор цикла наилучшим образом подходит для обработки всех элементов массива?

1. for
2. while.
3. foreach.
4. do...while.

10. Какая функция PHP используется для открытия файла?

1. open_file()
2. fclose().
3. open().
4. fopen().

11. Какая функция PHP используется для удаления файла?

1. del().
2. unlink()
3. move().
4. unset().

12. Какая функция PHP конвертирует символы новой строки в теги?

1. convert().
2. nl2br()
3. strip_tags().
4. htmlentities().

13. Какая функция PHP используется для чтения 1 символа из файла?

1. file().
2. fgetc()
3. fgets().
4. read().

14. Какой из операторов PHP позволяет определить остаток от целочисленного деления?

1. /.
2. \.
3. %
4. mod.

15. Когда после редактирования файла конфигурации PHP необходимо перезапустить web-сервер с тем, чтобы сделанные изменения вступили в силу?

1. Когда PHP установлен как CGI-приложение.
2. Когда PHP установлен как ISAPI-модуль.
3. Web-сервер необходимо перезапускать в любом случае
4. Для этих целей web-сервер перезапускать не требуется никогда.

16. С помощью, какой функции PHP можно изменить каталог, в котором сохраняются файлы с данными пользовательских сессий?

1. save_path().
2. save().
3. session_save_path()
4. изменить данный каталог можно только в файле конфигурации PHP.

17. С помощью какой функции PHP можно изменить текущий каталог?

1. mkdir().
2. rmdir().
3. chdir()
4. dir().

18. Какие имена переменных в PHP являются правильными?

1.$var1
2. var2.
3. $_var3.

19. Какой параметр конфигурации разрешает использовать в PHP теги <? ?>?

1. asp_tags.
2. short_open_tags
3. tags.

20. Какая функция PHP используется для установки позиции курсора в файле?

1. ftell()
2. fseek().
3. feof().
4. fputs().

6.4.3 Таблица ответов на вопросы рубежных контролей №1, №2. Очная форма обучения, 4 семестр

№ вопроса	Правильные ответы			
	Рубежный контроль №1		Рубежный контроль №2	
	Вариант 1_1	Вариант 1_2	Вариант 2_1	Вариант 2_2
1	1	1,2,3	1	1
2	4	1,4	3	1
3	2	2	2	1
4	1	1,3	2	4
5	3	1,4	4	1
6	4	4	1	2
7	3	3,4	3	3
8	2	1,3,4	2	4
9	4	1,2	1	1
10	2	1,3	3	4
11	1	2	1	2
12	1	3	2	2

13	3	2	4	2
14	2	1,3	2	3
15	1	2	2	3
16	3	2,4	2	3
17	4	1,3	1	3
18	2	1,2	3	1
19	1	2,5	3	2
20	3	1	5	2

6.4.4 Примеры заданий для рубежного контроля №1
Очная форма обучения, 5 семестр

Вариант 1_1

1 Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта?

*1 Да

2 Нет

2 Какие серверные программы пишутся разработчиком, добавляются во фреймворк и реализуют специфическую функциональность web-сайта?

1 Концентратор

2 Коммутатор

3 Адаптер

*4 Маршрутизатор

3 Какие фреймворки относятся к PHP-фреймворкам?

1 Pyramid

*2 Phalcon

3 Sinatra

4 Tomado

4 Какие требования предъявляются PHP, под котором Laravel будет работать?

1 Расширение PHP OpenLSS

2 Расширение PHP Nbstring

*3 Расширение PHP PDO

4 Расширение PHP BB Math

5 Какие файлы содержатся в папке проекта?

1 ertisan

2 ortisan

*3 artisan

4 irtisan

6 Какой правильный программный код новой миграции?

1 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}
}
public function down() {}
}

*2 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Schema\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}
}
public function down() {}
}

3 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Fasades\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends R
{
public function up() {}
}
public function down() {}
}

4 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Migrations\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}
}
public function down() {}
}

7 Какой правильный алгоритм создания связей между таблицами?

1

1 Этап. Создаётся первичная таблица.
2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для первичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onUpdate*.

2

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями первичной таблицы при удалении связанной с ними записи вторичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onUpdate*.

*3

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Blueprint*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первич-

ной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Blueprint* метод *onUpdate*.

4

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onUpdate.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onDelete*.

8 Какие правильные требования предъявляются модели Laravel?

*1

1 Класс модели должен иметь имя, совпадающее с именем сущности, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем класса.

2

1 Класс модели должен иметь имя, совпадающее с именем сущности, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем сущности.

3

1 Класс модели должен иметь имя, совпадающее с именем класса, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем класса.

4

1 Класс модели должен иметь имя, совпадающее с именем класса, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем сущности.

9 Какие правильные соглашения по умолчанию должны выполняться для моделей Laravel?

1

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *config* папки проекта.

2

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *vendor* папки проекта.

*3

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *app* папки проекта.

4

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *.env* папки проекта.

10 Какой правильный алгоритм создания сквозной связи?

1

1 Этап. В классе модели, относящейся ко вторичной таблице, объявляется открытый метод, имя которого совпадает с именем первичной таблицы и который не должен принимать параметров.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

2

1 Этап. В классе модели, относящейся к первичной таблице, объявляется открытый метод, имя которого совпадает с именем вторичной таблицы и который должен принимать параметры.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

3

1 Этап. В классе модели, относящейся ко вторичной таблице, объявляется открытый метод, имя которого совпадает с именем первичной таблицы и который должен принимать параметры.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

*4

1 Этап. В классе модели, относящейся к первичной таблице, объявляется открытый метод, имя которого совпадает с именем вторичной таблицы и который не должен принимать параметров.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

11 В каких файлах хранятся настройки маршрутизации?

*1 web.php

2 conf.php

*3 api.php

4 doc.php

12 Какие методы фасада Route используются для указания маршрутов?

1 set()

*2 get()

3 path()

*4 patch()

13 Какие скобки используются для обозначения получаемого параметра в параметризованном маршруте?

1 ()

2 []

*3 {}

4 <>

14 Именованный маршрут – маршрут, имеющий имя, по которому на него можно будет сослаться

*1 Да

2 Нет

15 Какой метод фасада Route создаёт посредников для маршрутов?

1 delete()

*2 middleware()

3 options{}

4 post()

16 Какой метод фасада Route используется для массового создания маршрутов?

1 many()

2 mass()

3 options()

*4 resources()

17 Каким образом осуществляется неявное внедрение модели в контроллер?

*1

1 Этап. Имя параметра в маршруте должно совпадать с именем соответствующего класса модели.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – класс соответствующей модели.

2

1 Этап. Имя параметра в маршруте должно совпадать с именем соответствующего класса модели.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – имя параметра маршрута.

3

1 Этап. Имя параметра в маршруте должно совпадать с именем папки проекта.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – класс соответствующей модели.

4

1 Этап. Имя параметра в маршруте должно совпадать с именем папки проекта.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – имя параметра маршрута.

18 Каким образом осуществляется явное внедрение модели в контроллер?

1 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: foot();
Route::model(<имя параметра>, <имя класса модели>);}
...
}
* 2 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: boot();
Route::model(<имя параметра>, <имя класса модели>);}
...
}
3 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: root();
Route::model(<имя параметра>, <имя класса модели>);}
...
}
4 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: poot();
Route::model(<имя параметра>, <имя класса модели>);}
...
}

19 Какие требования предъявляются контроллеру Laravel?

1

- Класс контроллера должен иметь имя, совпадающее с именем папки проекта, в которой он размещён, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

*2

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

3

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

4

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем сущности.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

20 Каким образом создаётся файл с классом контроллера?

1 *php artisan make:controller <имя контроллера> [--options]*

2 *php artisan make:controller <имя контроллера> [--source]*

*3 *php artisan make:controller <имя контроллера> [--resource]*

4 *php artisan make:controller <имя контроллера> [--reqiare]*

Вариант 1_2

1 Какой правильный алгоритм создания сквозной связи?

1

1 Этап. В классе модели, относящейся ко вторичной таблице, объявляется открытый метод, имя которого совпадает с именем первичной таблицы и который не должен принимать параметров.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

2

1 Этап. В классе модели, относящейся к первичной таблице, объявляется открытый метод, имя которого совпадает с именем вторичной таблицы и который должен принимать параметры.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

3

1 Этап. В классе модели, относящейся ко вторичной таблице, объявляется открытый метод, имя которого совпадает с именем первичной таблицы и который должен принимать параметры.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

*4

1 Этап. В классе модели, относящейся к первичной таблице, объявляется открытый метод, имя которого совпадает с именем вторичной таблицы и который не должен принимать параметров.

2 Этап. В теле объявленного на 1 этапе метода вызывается метод *hasManyThrough* класса *Model*.

2 Какие скобки используются для обозначения получаемого параметра в параметризованном маршруте?

1 ()

2 []

*3 {}
4 <>

3 Какой метод фасада Route создаёт посредников для маршрутов?

1 delete()
*2 middleware()
3 options{}
4 post()

4 Каким образом осуществляется неявное внедрение модели в контроллер?

*1

1 Этап. Имя параметра в маршруте должно совпадать с именем соответствующего класса модели.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – класс соответствующей модели.

2

1 Этап. Имя параметра в маршруте должно совпадать с именем соответствующего класса модели.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – имя параметра маршрута.

3

1 Этап. Имя параметра в маршруте должно совпадать с именем папки проекта.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – класс соответствующей модели.

4

1 Этап. Имя параметра в маршруте должно совпадать с именем папки проекта.

2 Этап. Имя параметра в объявлении метода-действия контроллера должно совпадать с именем параметра маршрута.

3 Этап. Для параметра в объявлении метода-действия контроллера должен быть указан тип данных – имя параметра маршрута.

5 Какие требования предъявляются контроллеру Laravel?

1

- Класс контроллера должен иметь имя, совпадающее с именем папки проекта, в которой он размещён, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

*2

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

3

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем класса.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

4

- Класс контроллера должен иметь имя, совпадающее с именем сущности, набор которых он обрабатывает, а завершаться его имя должно словом *Controller*.

- Класс контроллера должен быть потомком класса *Controller* или *BaseController*.

- Классы контроллера должны находиться в пространстве имён *app\Http\Controllers* или вложенном в него.

- Код класса контроллера должен храниться в файле с именем, совпадающем с именем сущности.

- Все файлы контроллера должны находиться в папке *app\Http\Controllers* или вложенной в неё папке, если используется вложенное пространство имён.

- Методы, реализующие действия, должны быть *public*.

6 Какие методы фасада Route используются для указания маршрутов?

1 set()
*2 get()
3 path()
*4 patch()

7 Каким образом создаётся файл с классом контроллера?

1 *php artisan make:controller <имя контроллера> [--options]*
2 *php artisan make:controller <имя контроллера> [--source]*
*3 *php artisan make:controller <имя контроллера> [--resource]*
4 *php artisan make:controller <имя контроллера> [--reqiare]*

8 В каких файлах хранятся настройки маршрутизации?

*1 web.php
2 conf.php
*3 api.php
4 doc.php

9 Каким образом осуществляется явное внедрение модели в контроллер?

1 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: foot();
Route::model(<имя параметра>, <имя класса модели>);}
...
}
* 2 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: boot();
Route::model(<имя параметра>, <имя класса модели>);}
...
}
3 *use App\Article;*
class RouteServiceProvider extends ServiceProvider {
...
public function boot()
{ parent: root();

Route::model(<имя параметра>, <имя класса модели>);}

...

}

4 *use App\Article;*

class RouteServiceProvider extends ServiceProvider {

...

public function boot()

{ parent: poot();

Route::model(<имя параметра>, <имя класса модели>);}

...

}

10 Какой метод фасада Route используется для массового создания маршрутов?

1 many()

2 mass()

3 options()

*4 resources()

11 Именованный маршрут – маршрут, имеющий имя, по которому на него можно будет сослаться

*1 Да

2 Нет

12 Какие серверные программы пишутся разработчиком, добавляются во фреймворк и реализуют специфическую функциональность web-сайта?

1 Концентратор

2 Коммутатор

3 Адаптер

*4 Маршрутизатор

13 Какие правильные требования предъявляются модели Laravel?

*1

1 Класс модели должен иметь имя, совпадающее с именем сущности, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем класса.

2

1 Класс модели должен иметь имя, совпадающее с именем сущности, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем сущности.

3

1 Класс модели должен иметь имя, совпадающее с именем класса, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем класса.

4

1 Класс модели должен иметь имя, совпадающее с именем класса, набором которых она манипулирует.

2 Класс модели должен быть потомком класса *Model* или его потомка.

3 Код класса модели должен храниться в файле с именем , совпадающим с именем сущности.

14 Какие правильные соглашения по умолчанию должны выполняться для моделей Laravel?

1

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *config* папки проекта.

2

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *vendor* папки проекта.

*3

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *app* папки проекта.

4

1 Таблица базы данных, с которой взаимодействует модель, имеет имя, совпадающее с именем модели, но записанное во множественном числе.

2 Таблица с которой взаимодействует модель, имеет служебные поля со следующими именами:

- ключевое поле *id*;
- поле для хранения даты и времени создания записи *created_at*;
- поле для хранения даты и времени последнего изменения записи *updated_at*;

3 Файлы с моделями находятся непосредственно в папке *.env* папки проекта.

15 Какие файлы содержатся в папке проекта?

1 ertisan
2 ortisan
*3 artisan
4 irtisan

16 Какие требования предъявляются PHP, под котором Laravel будет работать?

1 Расширение PHP OpenLSS
2 Расширение PHP Nbstring
*3 Расширение PHP PDO
4 Расширение PHP BB Math

17 Фреймворк – программная платформа, определяющая структуру программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта?

*1 Да
2 Нет

18 Какой правильный программный код новой миграции?

1 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}
}
public function down() {}
}
2 <?php
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Schema\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}

}
public function down() {}
}
3 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Fasades\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends R
{
public function up() {}
}
public function down() {}
}
4 *<?php*
use Illuminate\Support\Fasades\Schema;
use Illuminate\Database|Migrations\Blueprint;
uae Illuminate\Database\Migrations\Migration;
class CreateCategoriesTable extends Migration
{
public function up() {}
}
public function down() {}
}

19 Какой правильный алгоритм создания связей между таблицами?

1

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для первичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onUpdate*.

2

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями первичной таблицы при удалении связанной с ними записи вторичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onUpdate*.

*3

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Blueprint*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onDelete.*

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Blueprint* метод *onUpdate*.

4

1 Этап. Создаётся первичная таблица.

2 Этап. Создаётся миграция для вторичной таблицы.

3 Этап. В миграции для вторичной таблицы указывается создание внешнего ключа на основе поля, по которому будет устанавливаться связь. Для этого применяется метод *foreign* класса *Blueprint*.

4 Этап. У объекта класса *Fluent,* возвращённого методом *foreign,* вызывается метод *references.*

5 Этап. У объекта класса *Fluent*, возвращённого методом *references,* вызывается метод *on*.

6 Этап. Если требуется указать, какие действия будут выполняться с записями вторичной таблицы при удалении связанной с ними записи первичной таблицы, следует вызывать у возвращённого методом *on* объекта *Fluent* метод *onUpdate*.

7 Этап. Если требуется указать, какие действия будут выполняться сполем внешнего ключа у записей вторичной таблицы при изменении значения ключевого поля в связанной записи первичной таблицы, следует вызвать у возвращённого методом *on* объекта класса *Fluent* метод *onDelete*.

20 Какие фреймворки относятся к PHP-фреймворкам?

1 Pyramid
*2 Phalcon
3 Sinatra
4 Tomado

6.4.5 Примеры заданий для рубежного контроля №2
Очная форма обучения, 5 семестр

Вариант 2_1

1 Какие виды контроллеров поддерживает Laravel?

1 Контроллеры-шаблоны
2 Контроллеры-маршруты
*3 Контроллеры-функции
*4 Контроллеры-действия

2 Какие методы выполняют фильтрацию по наличию или отсутствию связанных записей?

*1 has()
2 haveDoesnt()
3 haveDoesntWhere()
4 orWhereHas()

3 Пагинатор – встроенный инструмент фреймворка Laravel для разбиения записей на страницы?

*1 Да
2 Нет

4 В чём отличие класса LengthAwarePaginator от класса Paginator?

1 Поддержка метода count()
*2 Поддержка метода total()
*3 Поддержка метода lastPage()
4 Поддержка метода isEmpty()

5 Какие методы класса ControllerMiddleware ограничивают перечень действий, подпадающих под влияние посредника?

1 firstItem()
2 perPage()
*3 only()
*4 except()

6 Шаблон – web-страница, в код которой вставлены инструкции, выполняющие вывод данных?

*1 Да

2 Нет

7 Какие требования и соглашения предъявляются шаблонам Laravel?

1

- Файлы с кодом шаблонов должны иметь blabe.php
- Файлы с шаблонами должны находиться в папке resources\views папки проекта или вложенной в нее папке.
- Помещать файлы шаблонов в папки, вложенные в папку resources\views, имена которых совпадают с именами контроллеров, использующих их для вывода данных.
- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

*2

- Файлы с кодом шаблонов должны иметь расширение blade.php
- Файлы с шаблонами должны находиться в папке resources\views папки проекта или вложенной в нее папке.
- Помещать файлы шаблонов в папки, вложенные в папку resources\views, имена которых совпадают с именами контроллеров, использующих их для вывода данных.
- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

3

- Файлы с кодом шаблонов должны иметь расширение blade.php
- Файлы с шаблонами должны находиться в папке resources папки проекта или вложенной в нее папке.
- Помещать файлы шаблонов в папки, вложенные в папку resources, имена которых совпадают с именами контроллеров, использующих их для вывода данных.
- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

4

- Файлы с кодом шаблонов должны иметь расширение blabe.php
- Файлы с шаблонами должны находиться в папке resources папки проекта или вложенной в нее папке.
- Помещать файлы шаблонов в папки, вложенные в папку resources, имена которых совпадают с именами контроллеров, использующих их для вывода данных.
- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

8 Какая папка по соглашению должна содержать вложенные шаблоны Laravel?

1 current

2 full
*3 common
4 default

9 Какая директива используется для создания шаблона-родителя?

1 @extends
2 @field
3 @tield
*4 @yield

10 Какая директива используется для создания шаблона-потомка?

*1 @extends
2 @field
3 @tield
4 @yield

11 Валидатор – инструмент Laravel для проверки данных?

*1 Да
2 Нет

12 Какие виды валидации используются в Laravel?

1 Комбинированная
*2 Автоматическая
*3 Полуавтоматическая
4 Ручная

13 Какие программные модули включает встроенная подсистема разграничения доступа Laravel?

*1 Контроллеры, находящихся в пространстве имён App\Http\Controllers\Auth
*2 Модель User, представляющую зарегистрированного пользователя
*3 Две миграции <дата и время>_create_users_table.php, <дата и время>_create_password_resets_table.php
4 Фасад Gate

14 Какие параметры поддерживаются библиотекой Captcha for Laravel?

1 lust
2 first
3 count
*4 invert

15 Laravel Mix – слой перед Webpack, упрощающий задачи и параметры конфигурации?

*1 Да
2 Нет

16 Кэширование на стороне сервера позволяет сохранить данные на дисках серверного компьютера и использовать их?

*1 Да
2 Нет

17 Какие хранилища позволяет использовать Laravel для хранения кэшированных данных?

*1 Папку
*2 Таблицу базы данных
*3 Оперативную память компьютера
4 Программы Mem и Red

18 Какой метод фасада Cache сохраняет в кэше несколько значений?

1 forever()
2 add()
*3 putMany()
4 pull()

19 Какой метод фасада Cache получает данные из кэша с последующим удалением?

1 forever()
2 add()
3 putMany()
*4 pull()

20 Какой метод фасада Cache j,hfoftncz r lheujve [hfybkboe?

1 get()
*2 store()
3 remember()
4 all()

Вариант 2_2

1 Валидатор – инструмент Laravel для проверки данных?

*1 Да
2 Нет

2 Какие виды валидации используются в Laravel?

1 Комбинированная
*2 Автоматическая
*3 Полуавтоматическая
4 Ручная

3 Кэширование на стороне сервера позволяет сохранить данные на дисках серверного компьютера и использовать их?

*1 Да
2 Нет

4 Какой метод фасада Cache сохраняет в кэше несколько значений?

1 forever()
2 add()
*3 putMany()
4 pull()

5 Какие параметры поддерживаются библиотекой Captcha for Laravel?

1 lust
2 first

3 count
*4 invert

6 Какой метод фасада Cache j,hfoftncz r lheujve [hfybkboe?

1 get()
*2 store()
3 remember()
4 all()

7 Какой метод фасада Cache получает данные из кэша с последующим удалением?

1 forever()
2 add()
3 putMany()
*4 pull()

8 Какие хранилища позволяет использовать Laravel для хранения кэшированных данных?

*1 Папку
*2 Таблицу базы данных
*3 Оперативную память компьютера
4 Программы Mem и Red

9 Laravel Mix – слой перед Webpack, упрощающий задачи и параметры конфигурации?

*1 Да
2 Нет

10 Какие программные модули включает встроенная подсистема разграничения доступа Laravel?

*1 Контроллеры, находящихся в пространстве имён App\Http\Controllers\Auth
*2 Модель User, представляющую зарегистрированного пользователя
*3 Две миграции <дата и время>_create_users_table.php, <дата и время>_create_password_resets_table.php
4 Фасад Gate

11 Какая директива используется для создания шаблона-потомка?

*1 @extends
2 @field
3 @tield
4 @yield

12 Пагинатор – встроенный инструмент фреймворка Laravel для разбиения записей на страницы?

*1 Да
2 Нет

13 Шаблон – web-страница, в код которой вставлены инструкции, выполняющие вывод данных?

*1 Да
2 Нет

14 Какие методы выполняют фильтрацию по наличию или отсутствию связанных записей?

*1 has()

2 haveDoesnt()

3 haveDoesntWhere()

4 orWhereHas()

15 Какая директива используется для создания шаблона-родителя?

1 @extends

2 @field

3 @tield

*4 @yield

16 Какая папка по соглашению должна содержать вложенные шаблоны Laravel?

1 current

2 full

*3 common

4 default

17 Какие требования и соглашения предъявляются шаблонам Laravel?

1

- Файлы с кодом шаблонов должны иметь blabe.php

- Файлы с шаблонами должны находиться в папке resources\views папки проекта или вложенной в нее папке.

- Помещать файлы шаблонов в папки, вложенные в папку resources\views, имена которых совпадают с именами контроллеров, использующих их для вывода данных.

- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

*2

- Файлы с кодом шаблонов должны иметь расширение blade.php

- Файлы с шаблонами должны находиться в папке resources\views папки проекта или вложенной в нее папке.

- Помещать файлы шаблонов в папки, вложенные в папку resources\views, имена которых совпадают с именами контроллеров, использующих их для вывода данных.

- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

3

- Файлы с кодом шаблонов должны иметь расширение blade.php

- Файлы с шаблонами должны находиться в папке resources папки проекта или вложенной в нее папке.

- Помещать файлы шаблонов в папки, вложенные в папку resources, имена которых совпадают с именами контроллеров, использующих их для вывода данных.

- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

4

- Файлы с кодом шаблонов должны иметь расширение blabe.php

- Файлы с шаблонами должны находиться в папке resources папки проекта или вложенной в нее папке.

- Помещать файлы шаблонов в папки, вложенные в папку resources, имена которых совпадают с именами контроллеров, использующих их для вывода данных.

- Давать каждому файлу шаблона имя, совпадающее с именем действия, которое использует этот шаблон.

18 Какие виды контроллеров поддерживает Laravel?

1 Контроллеры-шаблоны

2 Контроллеры-маршруты

*3 Контроллеры-функции

*4 Контроллеры-действия

19 Какие методы класса ControllerMiddleware ограничивают перечень действий, подпадающих под влияние посредника?

1 firstItem()

2 perPage()

*3 only()

*4 except()

20 В чём отличие класса LengthAwarePaginator от класса Paginator?

1 Поддержка метода count()

*2 Поддержка метода total()

*3 Поддержка метода lastPage()

4 Поддержка метода isEmpty()

6.4.6 Таблица ответов на вопросы рубежных контролей №1, №2. Очная форма обучения, 5 семестр

№ вопроса	Правильные ответы			
	Рубежный контроль №1		Рубежный контроль №2	
	Вариант 1_1	Вариант 1_2	Вариант 2_1	Вариант 2_2
1	1	4	3,4	1
2	4	3	1	2,3
3	2	2	1	1
4	3	1	2,3	3
5	3	2	3,4	4
6	2	2,4	1	2
7	3	3	2	4
8	1	1,3	3	1,2,3
9	3	2	4	1

10	4	4	1	1,2,3
11	1,3	1	1	1
12	2,4	4	2,3	1
13	3	1	1,2,3	1
14	1	3	4	1
15	2	3	1	4
16	4	3	1	3
17	1	1	1,2,3	2
18	2	2	3	3,4,1
19	2	3	4	3,4
20	3	2	2	2,3

6.4.7 Примерный перечень вопросов для зачёта

1 Протоколы и модели Internet-взаимодействия. Концептуальная модель Web.

2 Ввод и форматирование текста в HTML.

3 Форматы графических файлов.

4 Графика, звук, мультимедиа на Web-страницах.

5 Связывание Web-страниц с помощью гиперссылок HTML.

6 Разметка Web-страниц с помощью таблиц в HTML.

7 Формы HTML. Элементы форм. Фокус ввода на элемент. Сценарии.

8 Фреймы в HTML. Определение, преимущества, недостатки. Создание структуры фреймов. Загрузка ссылок во фреймы.

9 Технология таблиц стилей. Включение CSS в HTML.

10 Синтаксис CSS. Правила, типы селекторов, единицы измерения, наследование, каскадирование.

11 Свойства CSS. Свойства управления цветом, шрифтами, текстом, таблицами.

12 Блоки в HTML и CSS.

13 Клиентские и серверные сценарии. Внедрение сценариев в JavaScript.

14 Программирование на JavaScript. Работа с данными в сценариях JavaScript.

15 Программирование на JavaScript. Управление ходом выполнения сценария JavaScript.

16 Встроенные объекты JavaScript: Global, String, Number, Boolean, Array, Function, Data, Math, RegExp, Object.

17 Родительские и дочерние объекты. Объекты браузера.

18 Объектная модель документа (Document Object Model).

19 JavaScript. Свойства, методы и события.

20 JavaScript. Элементы управления в формах.

21 JavaScript. Фреймы. Создание фреймов. Вложенные и плавающие фреймы.

22 JavaScript. Создание и применение составных окон. Ссылки на фреймы.

23 Управление текстом в динамическом HTML.

24 Графика, анимация, изображения в динамическом HTML.

25 Таблицы стилей. Способы использования стилей. Доступ к стилям с помощью JavaScript.

26 Отладка сценариев JavaScript. ScriptDebugger.

27 Cookies. Операции с cookies: создание, получение, изменение параметров, удаление.

28 Массивы в PHP. Синтаксис. Работа с массивами. Реализация массивов.

29 Работа с формами в PHP. Обработка и преобразование данных. Функции protected() и validate().

30 Реализация cookie-наборов в PHP.

31 Объектно-ориентированное программирование в PHP.

32 Использование PHP для создания консольных сценариев.

33 Работа с файлами в PHP.

34 Работа с данными в PHP. Использование MySQL в PHP.

35 Работа с изображениями в PHP.

36 Базовый синтаксис PHP. Типы данных, управляющие структуры, динамические переменные и функции.

6.4.8 Примерный перечень вопросов для экзамена

1. Динамические страницы и сайты. Преимущества и недостатки. Разграничение доступа.

2. Разработка серверных программ. Виды серверных программ. Преимущества и недостатки.

3 Фреймворки. Основные понятия и определения. Средства и инструменты фреймворков. Виды серверных программ. Особенности и характеристики. Виды фреймворков.

4 Установка и настройка фреймворка Laravel. Программные требования, Создание нового проекта. Структура папок Laravel-проекта. Настройки сайта: соединения с базой данных, отправки электронной почты, режима работы сайта.

5 Миграции Laravel. Создание миграций. Фасады Laravel. Создание структур данных. Правка и переименование структур данных. Удаление структур данных.

6 Модели Laravel. Требования и соглашения. Создание простых моделей. Создание связей. Расширение функциональности модели.

7 Паттерн MVC. HTTP команды. REST – архитектурный стиль для создания API. Определения маршрутов: команды маршрутов, обработка маршрутов, параметры маршрутов, имена маршрутов. Файлы хранения настроек маршрутизации.

8 Маршрутизация в Laravel. Создание маршрутов: простые, параметризованные, именованные. Массовое создание маршрутов. Внедрение модели в контроллер. Группы маршрутов. Подписанные маршруты.

9 Контроллеры Laravel. Требования и соглашения. Созддание контроллеров. Получение данных от посетителя.

10 Контроллеры Laravel. Работа с базой данных. Простая выборка ззаписей. Получение значений полей записи. Получение связанных записей. Создание запросов к базе данных. Использование агрегатных функций. Группировка записей. Использование пагинатора.

11 Контроллеры Laravel. Получение сведений о запросе. Получение путей к папкам фреймворка. Вывод данных: посредством шаблонов, в формате JSON, Перенаправление. Указание посредников в контроллере. Виды контроллеров.

12 Система объектно-реляционного отображения Eloquent. Создание и определение моделей Eloquent. Получение данных с помощью Eloquent. Вставки и обновления с помощью Eloquent.Удаление с помощью Eloquent. Коллекции Eloquent.Связи в Eloquent. События Eloquent.

13 Шаблоны Laravel. Требования и соглашения. Создание шаблонов. Язык шаблонов Blade. Отображение данных. Управляющие структуры: условные конструкции, циклы. Наследование шаблонов: определение разделов страницы с помощью директив @section, @show, @yield.

14 Шаблоны Laravel. Включение составляющих представления: использование стеков, использование компонентов и слотов. Компоновщики представлений и внедрение сервисов: привязка данных к представлениям с использованием компоновщиков представлений, внедрение сервиса Blade.

15 Шаблоны Laravel. Пользовательские директивы Blade: параметры пользовательских директив Blade,упрощённые пользовательские директивы для операторов if. Комментарии Blade.Вставка PHP-кода. Особые случаи вывода данных: генерирование интернет-адресов, создание web-форм и элементов управления, вывод всплывающих сообщений, вывод пагинатора.

16 Шаблоны Laravel. Вложенные шаблоны. Наследование шаблонов: создание шаблонов-родителей, создание шаблонов-потомков. Стеки. Разделяемые данные и составители. Получение доступа к контроллеру.

17 Laravel Mix. Структура каталога Mix. Запуск Mix. Предустановки клиентской части и генерация кода аутентификации. Разбивка на страницы: разбивка на страницы результатов из базы данных, создание разбивщиков страниц вручную.

18 Laravel Mix. Панель сообщений. Строковые хелперы, множественность и локализация: строковые хелперы и множественность, локализация. Тестирование: тестирование пакетов сообщений и ошибок, перевод и локализация.

19 Встроенная система разграничения доступа Laravel. Ограничение доступа к страницам: простые случаи ограничения доступа, ограничение доступа на основе сложных условий (шлюзы, политики), ограничение доступа

на основе запросов форм, ограничение доступа в шаблонах. Вывод сведений о текущем пользователе.

20 Настройка встроенной подсистемы разграничения доступа: базовые настройки, модификация списка пользователей, настройка писем, отправляемых действием восстановления пароля (создание оповещения, подготовка шаблона для оповещения, отправка оповещений).

21 Использование CAPTCHA. Библиотека Captcha for Laravel: установка, настройка, использование.

22 Сохранение и извлечение данных. Локальные и облачные файловые менеджеры: настройка доступа к файлам, фасад Storage, пакет Flysystem. Базове способы загрузки файлов на сервер и манипулирование файлами.

23 Сохранение и извлечение данных. Простые способы скачивания файлов. Сессии: получение доступа к сессии, методы, доступные в экземплярах сессий, флеш-память сессии.

24 Сохранение и извлечение данных. Настройки кэширования: кэширование в папке, кэширование в таблице базы данных, кэширование в оперативной памяти. Занесение данных в кэш. Изменение данных, хранящихся в кэше. Сохранение и извлечение данных. Получение данных из кэша. Удаление данных из кэша.

25 Сохранение и извлечение данных. Cookie-файлы: cookie-файлы в Laravel, получение доступа к cookie-файлам. Логирование: внесение записей в логи, каналы логирования.

26 Сохранение и извлечение данных. Полнотекстовый поиск с использованием Laravel Scout: пакет Scout, пометка модели для индексирования, поиск по индексу, очереди и Scout, выполнение операций без индексирования, условное индексирование моделей, запуск индексирования вручную с помощью кода, запуск индексирования вручную с помощью интерфейса командной строки.

27 Ввод и правка данных. Создание, правка и удаление отдельных записей: создание web-формы для ввода и правки записи, создание записи, правка записи, удаление записи, обработка связей между таблицами.

28 Ввод и правка данных. Дополнительные инструменты для создания и правки отдельных записей: поиск или создание записей, исправление или создание записей, работа со связанными записями. Проверка введённых в форму данных на корректность.

29 Ввод и правка данных. Валидаторы. Простейшие валидаторы: автоматическая валидация, полуавтоматическая валидация, условные правила ввалидации. Правила валидации.

30 Ввод и правка данных. Запросы форм. Массовое создание, правка и удаление записей. Работа с выгруженными файлами: файловое хранилище и диски laravel, создание web-формы для выгрузки файлов, получение и сохранение выгруженных файлов, работа с выгруженными файлами.

6.4.9 Контрольная работа

6.4.9.1 Назначение, цели и задачи контрольной работы

Контрольная работа выполняется по вариантам заданий или по теме, предложенной студентом, и согласованной с преподавателем.

В ходе выполнения контрольной работы студент проектирует и реализует полнофункциональный сайт электронной коммерции.

Основная учебная цель выполнения контрольной работы – закрепление теоретических знаний, полученных в процессе изучения дисциплины «Технологии разработки web-приложений», и приобретение практических навыков в разработке полнофункционального сайта.

Основные задачи, решаемые студентом в процессе выполнения контрольной работы:

- выбрать технологии реализации Internet сценария;
- осуществить поиск информации для наполнения контента сайта;
- спроектировать структуру сайта;
- спроектировать базу данных;
- создать классы;
- разработать Internet сценарий с одной или несколькими функциональными возможностями по выбору:

1. Система администрирования контента сайта с возможностями
   - изменять и корректировать структуру сайта;
   - редактировать существующие страницы и создавать новые;
   - редактировать информационное наполнение страниц сайта.
2. Универсальный каталог продукции.
3. Система полнотекстового поиска по сайту.
4. Представление контактной информации, включающей
   - систему представления информации;
   - систему администрирования.
5. Представление системы голосования.
6. Гостевая книга, включающая
   - блок представления страниц для добавления и отображения сообщений на сайте;
   - блок администрирования страниц, где администратор может осуществить модерирование сообщений и ответить на вопросы посетителей.
7. Система публикаций сообщений на сайте – «Фотогаллерея».
8. Система отправки почтовых сообщений с сайта.
9. Система анализа посещаемости сайта – Power Counter:
   - общее число хитов и хостов для сайта в целом и каждой страницы в отдельности;
   - число посещений с поисковых систем и других ресурсов, где расположена ссылка на сайт.
   - IP-адреса и время посещения сайта посетителем;

-ключевые слова, по которым сайт был найден в поисковых системах;

- количество роботов, посещающих сайт;
- тип и время операционной системы и браузера посетителя и др.;

10. Система мониторинга позиций сайта в поисковых системах.

11. Web-приложение, обеспечивающее доступ по HTTP к удаленному серверу FTP, с другого удаленного сервера к локальной ПЭВМ

- свободное перемещение по структуре FTP-сервера;
- представление информации о владельцах, группах, обладающих правами доступа к файлам и директориям;
- создание, переименование, удаление и изменение прав доступа к директориям;
- загрузка на сервер и с сервера файлов, переименование и изменение прав доступа.

12. Web-приложение «Почтовая служба»

- создание учетных записей и подключение к ним;
- чтение сообщений электронной почты;
- отправка сообщений электронной почты;
- удаление сообщений электронной почты;
- переадресация сообщений электронной почты.

13. Система рассылки писем.

14. Форум.

- оформить документацию.

6.4.9.2 Требования к контрольной работе

6.4.9.2.1 Требования к функциональным характеристикам

Проектируемая система должна обеспечивать выполнение следующих основных функций:

- авторизация пользователя;
- администрирование контента сайта;
- универсальный каталог;
- поиск по ключевым словам;
- система голосования;
- гостевая книга;
- система публикаций сообщений;
- система отправки почтовых сообщений с сайта;
- система анализа посещаемости сайта;
- форум.
- хранение данных;
- вывод результирующих данных.

6.4.9.2.2 Требования к эксплуатационным характеристикам

- Модульность.
- Расширяемость.

6.4.9.2.3 Требования к программному обеспечению

- Объектно-ориентированный язык программирования PHP версии не ниже 5.0;
- СУБД MySQL версии не ниже 5.0.

6.4.9.2.4 Требования к содержанию контрольной работы

К защите контрольной работы должен быть представлен альбом, включающий следующие проектные, программные и эксплуатационные документы:

- Опись альбома.
- Пояснительная записка, включающая разделы:
- аналитический обзор;
- описание алгоритмов решения задачи;
- описание структуры программного комплекса;
- описание структур данных;
- схема данных;
- описание Internet сценария.
- Спецификация.
- Описание программы.
- Текст программы (на машинном носителе).
- Руководство пользователя.
- Руководство программиста.

Пояснительная записка оформляется в соответствии с требованиями методического указания к оформлению документации курсовых и дипломных проектов.

6.4.9.3 Варианты заданий контрольной работы

1 Фирма, осуществляющая продажу средств связи.
2 Фирма, осуществляющая продажу легковых автомобилей.
3 Фирма, предоставляющая риэлтерские услуги.
4 Фирма, предоставляющая туристические услуги.
5 Фирма, осуществляющая продажу строительных материалов.
6 Фирма, предоставляющая юридические услуги.
7 Фирма, осуществляющая продажу торгового оборудования.
8 Фирма, осуществляющая продажу книгопечатной продукции.
9 Фирма, осуществляющая продажу лакокрасочной продукции.
10 Фирма, осуществляющая продажу продуктов питания.

11 Фирма, осуществляющая продажу вычислительной техники.
12 Фирма, осуществляющая продажу теле-, видеоаппаратуры.
13 Фирма, осуществляющая продажу косметики.
14 Фирма, осуществляющая продажу музыкальных инструментов.
15 Фирма, осуществляющая продажу вело-, мототехники.
16 Фирма, осуществляющая продажу кабельного оборудования.
17 Фирма, осуществляющая продажу спутникового оборудования.
18 Фирма, осуществляющая продажу электротоваров.
19 Фирма, осуществляющая продажу сельскохозяйственной техники.
20 Фирма, осуществляющая продажу канцелярских принадлежностей.

6.4.10 Курсовой проект

6.4.10.1 Назначение, цели и задачи курсового проекта

Курсовой проект выполняется по вариантам заданий или по теме, предложенной студентом, и согласованной с преподавателем.

В ходе выполнения курсового проекта студент проектирует и реализует сайт с использованием фреймворка Laravel.

Основная учебная цель выполнения курсового проекта– закрепление теоретических знаний, полученных в процессе изучения дисциплины «Технологии разработки web-приложений», и приобретение практических навыков в разработке сайта с использованием фреймворка Laravel.

Основные задачи, решаемые студентом в процессе выполнения курсового проекта:

- выбрать технологии реализации web-сайта;
- осуществить поиск информации для наполнения контента сайта;
- спланировать web-сайт;
- спроектировать базу данных;
- создать дизайн web-страниц;
- создать интерактивные элементы web-сайта;
- создать статические web-страницы;
- выполнить разграничение доступа и создать список пользователей;
- разработать страницы для управления категориями и подкатегориями;
- разработать вывод на экран перечня статей, относящихся к выбранной посетителем подкатегории;
- разработать вывод последних пяти статей, относящихся к выбранной категории;
- реализовать возможность оставлять комментарии к статьям;
- реализовать хранение выгруженных пользователями файлов и вставку ссылок на них в текст статей;
- подготовить web-сайт к публикации.

6.4.10.2 Требования к курсовому проекту

6.4.10.2.1 Требования к функциональным характеристикам

Проектируемая система должна обеспечивать выполнение следующих основных функций:

- ввод и редактирование статей;
- размещение на мобильных устройствах;
- реализация спойлера, лайтбокса, блокнота;
- разграничение доступа и список пользователей;
- управление категориями и подкатегориями;
- вывод на экран перечня статей, относящихся к выбранной посетителем подкатегории;
- вывод последних пяти статей, относящихся к выбранной категории;
- комментарии к статьям;
- хранение выгруженных пользователями файлов и вставка ссылок на них в текст статей.

6.4.10.2.2 Требования к эксплуатационным характеристикам

- Модульность.
- Расширяемость.

6.4.10.2.3 Требования к программному обеспечению

- Система WAMP Open Server версии 5.3.7 и новее;
- Web-сервер Apache версии 2.4 и новее;
- Объектно-ориентированный язык программирования PHP версии не ниже 5.6 и новее;
- СУБД MySQL версии не ниже 5.5 и новее.

6.4.10.2.4 Требования к содержанию курсового проекта

К защите курсового проекта должны быть представлены полнофункциональный сайт и альбом, включающий документы:

- Опись альбома.
- Пояснительная записка, содержащая разделы:
- Концепция веб-сайта
- Архитектура веб-сайта
- Обоснование выбора программного обеспечения
- Информационное наполнение веб-сайта
- Структура веб-сайта
- Проектирование базы данных
- Реализация веб-сайта
- Спецификация.

- Описание программы.
- Текст программы (на машинном носителе).
- Руководство пользователя.
- Руководство программиста.

6.4.10.3 Примерный перечень тем курсового проекта

Вариант 1

1 Web-сайт электронной публикации научных статей для университета “МГТУ имени Н.Э. Баумана”.

2 Web-сайт электронной публикации статей для детского журнала “Электроник”.

3 Web-сайт электронной публикации статей для научного журнала “Наука и жизнь”

4 Web-сайт электронной публикации статей для спортивного журнала “Спорт”

5 Web-сайт электронной публикации статей для общества любителей путешествий “Вокруг света”.

6 Web-сайт электронной публикации статей общества любителей природы “Мир природы”.

7 Web-сайт электронной публикации статей для журнала “Техника молодёжи”.

8 Web-сайт электронной публикации статей для журнала “Здоровье”.

9 Web-сайт электронной публикации статей для журнала “Футбол”.

10 Web-сайт электронной публикации статей для журнала “Охота Рыбалка”.

11 Web-сайт электронной публикации статей для журнала “Финансовый бизнес”.

12 Web-сайт электронной публикации статей для журнала “Автомир”.

13 Web-сайт электронной публикации статей для журнала “Военно-стратегический анализ”.

14 Web-сайт электронной публикации статей для журнала “Великая Отечественная”.

15 Web-сайт электронной публикации статей для журнала “Computer World”.

16 Web-сайт электронной публикации статей для журнала “Добрые советы”.

17 Web-сайт электронной публикации статей для журнала “Наша кухня”.

18 Web-сайт электронной публикации статей для журнала “Смена”.

19 Web-сайт электронной публикации статей для журнала “Делаем сами”.

20 Web-сайт электронной публикации статей для журнала “Аэрокосмическое обозрение”.

Вариант 2

1 Web-сайт по продаже садовых растений для компании “Сад”.

2 Web-сайт по продаже легковых автомобилей для компании “Авто”.

3 Web-сайт по продаже вело-мототехники для компании “Мототехника”.

4 Web-сайт по продаже музыкальных инструментов для компании “Муза”.

5 Web-сайт по продаже электротоваров для компании “Электрон”.

6 Web-сайт по продаже спутникового оборудования для компании “Спутник”.

7 Web-сайт по продаже строительных материалов для компании “Стройматериал”.

8 Web-сайт по продаже сельскохозяйственной техники для компании “Нива”.

9 Web-сайт по продаже лакокрасочной продукции для компании “Лак”.

10 Web-сайт по продаже вычислительной техники для компании “Компьютер”.

11 Web-сайт по продаже мебельной продукции для компании “Мебель”.

12 Web-сайт по продаже книгопечатной продукции для компании “Книга”.

13 Web-сайт по продаже парфюмерной продукции для компании “Парфюм”.

14 Web-сайт по продаже канцелярских принадлежностей для компании “Карандаш”.

15 Web-сайт по продаже средств связи для компании “Радио”.

16 Web-сайт по продаже кабельного оборудования для компании “Кабель”.

17 Web-сайт по продаже ювелирных изделий для компании “Изумруд”.

18 Web-сайт по продаже плодовоовощной продукции для компании “Фрукты-овощи”.

19 Web-сайт по продаже автозапчастей для компании “Автозапчасти”.

20 Web-сайт по продаже недвижимости для компании “Дом”.

Вариант 3

1. Разработка информационного сайта.
2. Разработка развлекательного сайта.
3. Разработка сайта презентации предприятия.
4. Разработка сайта дистанционного обучения.
5. Разработка рекрутингового сайта.
6. Разработка сайта-визитки.
7. Разработка корпоративного информационного web-сайта.
8. Разработка корпоративного имиджевого web- сайта.

9. Разработка игрового портала.
10. Разработка персонального проекта.
11. Разработка контент-проекта.
12. Разработка промосайта.
13. Разработка сайт-форума.
14. Разработка блога.
15. Разработка информационного сайта.
16. Разработка корпоративного имиджевого web- сайта.
17. Разработка корпоративного информационного web-сайта.
18. Разработка сайта дистанционного обучения.
19. Разработка рекрутингового сайта.
20. Разработка развлекательного сайта.

6.5. Фонд оценочных средств

Полный банк заданий для текущего, рубежных контролей и промежуточной аттестации по дисциплине, показатели, критерии, шкалы оценивания компетенций, методические материалы, определяющие процедуры оценивания образовательных результатов, приведены в учебно-методическом комплексе дисциплины.

7. ОСНОВНАЯ И ДОПОЛНИТЕЛЬНАЯ УЧЕБНАЯ ЛИТЕРАТУРА

7.1. Основная учебная литература

1 Стаффер Мэтт. Laravel. Полное руководство. 2-е изд. – СПб.: Питер, 2020. – 512 с

2 Прохоренок Н. А. HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера. – 5-е изд., перераб. и доп. / Н. А. Прохоренок, В. А. Дронов. – СПб.: БХВ-Петербург, 2019. – 912 с.

3 Дронов В. А. Laravel. Быстрая разработка современных динамических Web-сайтов на PHP, MySQL, HTML и CSS. – СПб.: БХВ-Петербург, 2018. – 768 с.

4 Никсон Р. Создаём динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML. 4-е изд. – СПб.: Питер, 2018. – 768 с.

5 Дронов В. А. PHP, MySQL, HTML 5 и CSS 3. Разработка современных динамических Web-сайтов. – СПб.: BHV-СПб, 2015 – 688 с.

6 Флэнаган Д. JavaScript. Подробное руководство. – М: Символ Плюс, 2012 – 1080 с.

7.2. Дополнительная учебная литература

1. Бенкен Е. С. PHP, MySQL, XML: программирование для Интернета. – СПб.: БХВ-Петербург, 2011. – 304 с.

2. Бенкен Е. С. AJAX: программирование для Интернета – СПб.: БХВ-Петербург, 2009. – 464 с.

3. Гольцман В. MySQL 5.0. Библиотека программиста. – СПб.: Питер, 2010. – 256 с.

4. Росс В.С. Создание сайтов: HTML, CSS, PHP, MySQL. Учебное пособие. Часть 1. - М.: МГДД(Ю)Т, 2010 - 107 с.

http://window.edu.ru/resource/489/69489 (дата обращения: 13.12.2020 г.).

8. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

1. Семахин А.М. Технологии разработки Web-приложений. Методические указания к выполнению лабораторных работ для студентов направления подготовки 09.03.03 «Прикладная информатика», 09.03.04 «Программная инженерия». Курган, КГУ, 2020. – 40 с. (электронный)

2. Семахин А.М. Технологии разработки Web-приложений. Методические указания к выполнению практических работ для студентов направления подготовки 09.03.03 «Прикладная информатика», 09.03.04 «Программная инженерия». Курган, КГУ, 2020. – 36 с. (электронный)

3 Семахин А.М. Технологии разработки Web-приложений. Методические указания к выполнению контрольных работ и курсового проекта для студентов направления подготовки 09.03.03 «Прикладная информатика», 09.03.04 «Программная инженерия». Курган, КГУ, 2020. – 38 с. (электронный).

4. Семахин А.М. Технологии разработки Web приложений учебное пособие. – Курган : Изд-во КГУ, 2020 – 54 с. (электронный).

9. РЕСУРСЫ СЕТИ «ИНТЕРНЕТ», НЕОБХОДИМЫЕ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1. Федеральный портал «Российское образование» URL: http://www.edu.ru/

2. Сайт дистанционного обучения в НОУ «ИНТУИТ». URL: http://www.intuit.ru/

10. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ И ИНФОРМАЦИОННЫЕ СПРАВОЧНЫЕ СИСТЕМЫ

При чтении лекций используются слайдовые презентации.

Минимальные требования к операционной системе и программному обеспечению компьютера, используемого при показе слайдовых презентаций: Windows XP, Foxit PDF Reader 12.1, 13.12.2022 г., свободное программное обеспечение (free software), GNU License.

11. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Материально-техническое обеспечение включает в себя учебные лаборатории и классы, оснащенные современными компьютерами (рабочими станциями локальной вычислительной сети) с доступом в Интернет, мультимедийное оборудование (переносной персональный компьютер, мультимедийный проектор, мультимедийный экран).

Программные средства обеспечения учебного процесса включают лицензионное программное обеспечение: операционную систему Windows XP, язык гипертекстовой разметки HTML, свободное (free software) и открытое (open source) программное обеспечение, объектно-ориентированный язык сценариев JavaScript (ECMAScript 2022), версии 9.0, июнь 2022 г. GNU License, web-сервер Apache 2.4.46, 05.08.2020 г., свободное (free software) программное обеспечение, Apache License 2.0, январь 2004 г., СУБД MySQL 8.0.32, 17.01.2023 г. свободное (free software) и открытое (open source) программное обеспечение, GNU General Public License, объектно-ориентированный язык PHP 8.2.4, 16.03.2023г., открытый код (open source), собственная лицензия, PHP License, фреймворк Laravel 9.0, 08.02.2022 г. свободное (free software) и открытое (open source) программное обеспечение, X11 License.

12. ДЛЯ ОБУЧАЮЩИХСЯ С ИСПОЛЬЗОВАНИЕМ ДИСТАНЦИОННЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

При использовании электронного обучения и дистанционных образовательных технологий (далее ЭО и ДОТ) занятия полностью или частично проводятся в режиме онлайн. Объем дисциплины и распределение нагрузки по видам работ соответствует п. 4.1. Распределение баллов соответствует п. 6.2 либо может быть изменено в соответствии с решением кафедры, в случае перехода на ЭО и ДОТ в процессе обучения. Решение кафедры об используемых технологиях и системе оценивания достижений обучающихся принимается с учетом мнения ведущего преподавателя и доводится до обучающихся.

ПРИЛОЖЕНИЕ 1

Аннотация к рабочей программе дисциплины

«Технологии разработки Web-приложений»

образовательной программы высшего образования –
программы бакалавриата

09.03.03 – Прикладная информатика
Направленность:
Интеллектуальные информационные системы и технологии

Трудоемкость дисциплины: 7 ЗЕ (252 академических часа)
Семестр: 4, 5 (очная форма обучения), 5, 6 (заочная форма обучения)
Форма промежуточной аттестации: зачёт, экзамен

Содержание дисциплины

Интернет-соединение. Протоколы. Адресация в сети Интернет.

Основные структурные элементы HTML-документа. Уровни заголовков. Шрифт. Абзацы и разрывы строк. Графические форматы (JPEG, GIF, PNG). Анимационные файловые форматы (SWF, AVI, MOV). Форматы MPEG. Звуковые форматы (WAV, AIFF, MIDI, MP3).

Создание таблиц. Создание форм. Теги. Свойства фреймов. Включение CSS в HTML. Клиентские и серверные сценарии. Переменные и типы данных. Выражения. Операторы. Функции и события. Встроенные объекты JavaScript. Свойства и методы объектов.

Объектная модель браузера и документа. Родительские и дочерние объекты. Объекты браузера. Объектная модель документа (Document Object Model, DOM). События динамического HTML. Принцип работы AJAX. Компоненты AJAX. Недостатки технологии AJAX.

Запросы к базе данных: команда SELECT, Редакторы для работы с PHP. Базовый синтаксис. Синтаксис XML. XML

Серверные программы. Фреймворки

Динамические страницы и сайты. Разработка серверных программ. История веб- и PHP-фреймворков: Ruby on Rails, Codelgniter, Laravel 1,2,3,4,5. Фреймворки: модели, шаблоны, контейнеры. Философия Laravel. Преимущества Laravel. Принцип работы Laravel. Сообщество Laravel.

Установка и настройка фреймворка Laravel. Миграции фреймворка Laravel. Модели фреймворка Laravel. Маршрутизация фреймворка Laravel. Контроллеры фреймворка Laravel. Шаблоны фреймворка Laravel. Ввод и правка данных. Компоненты для клиентской части. Разграничение доступа. Использование CAPTCHA. Сохранение и извлечение данных.