РОСЖЕЛДОР
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Ростовский государственный университет путей сообщения»
(ФГБОУ ВО РГУПС)

В.С. Якуничев

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ ПО ДИСЦИПЛИНЕ

МДК.04.01 «Бэкенд-разработка (серверная часть)»

для специальности
09.02.09 Веб-разработка

Ростов-на-Дону
2025

СОДЕРЖАНИЕ

1 САМОСТОЯТЕЛЬНАЯ РАБОТА СТУДЕНТОВ

Самостоятельная работа студентов является важнейшей составной частью образовательного процесса и способом активного, целенаправленного приобретения студентом новых знаний и умений без непосредственного участия преподавателя, но под его методическим руководством.

Изучение бэкенд-разработки требует глубокого понимания принципов работы веб-серверов, протоколов передачи данных и алгоритмического мышления. Самостоятельная работа позволяет закрепить навыки написания программного кода и разобраться в архитектурных особенностях веб-приложений.

Целями самостоятельной работы являются:

– систематизация и закрепление полученных теоретических знаний и практических умений в области серверного программирования;

– углубление и расширение теоретических знаний по синтаксису языка PHP и работе с протоколом HTTP;

– формирование умений работать с технической документацией и стандартами (PSR);

– развитие навыков отладки программного кода и обеспечения безопасности веб-ресурсов.

Организационные мероприятия, обеспечивающие нормальное функционирование самостоятельной работы студента:

– Самостоятельная работа должна быть конкретной по своей предметной направленности (решение прикладных задач).

– Самостоятельная работа должна сопровождаться эффективным, непрерывным контролем и оценкой ее результатов (код-ревью, защита лабораторных).

Виды самостоятельной работы по дисциплине «Бэкенд-разработка»:

– Работа с теоретическим материалом: изучение документации PHP (php.net), стандартов HTTP, принципов MVC.

– Подготовка к практическим занятиям: настройка локального веб-сервера (Denwer), написание базовых алгоритмов.

– Выполнение индивидуального задания (проекта): разработка динамического веб-сайта с системой администрирования на файловой базе данных (без использования SQL).

– Подготовка к зачету и экзамену: систематизация знаний по функционированию серверной части веб-приложений.

2 СТРУКТУРА ДИСЦИПЛИНЫ

Введение. Дисциплина «Бэкенд-разработка (серверная часть)» реализуется в учебном плане среднего профессионального образования по специальности 09.02.09 «Веб-разработка».

Место дисциплины в структуре образовательной программы

Дисциплина является логическим продолжением курсов «Основы веб-разработки и верстки» и «Фронтенд-разработка». Она закладывает фундамент для последующего изучения работы с базами данных и проектирования сложных информационных систем.

Дисциплина отнесена к профессиональному циклу Образовательной программы, реализуется в рамках профессионального модуля ПМ.04 «Разработка веб-приложения на стороне сервера».

Дисциплина реализуется в пятом и шестом семестре.

Форма контроля – зачет, экзамен.

Целью освоения дисциплины «Бэкенд-разработка (серверная часть)» является формирование у студентов системы знаний о принципах функционирования веб-серверов и интерпретаторов, а также выработка практических навыков разработки серверной логики веб-приложений с использованием языка программирования PHP, обеспечивающих взаимодействие с пользователем, обработку данных и генерацию динамического контента.

Задачи освоения дисциплины:

Изучение архитектурных принципов: формирование глубокого понимания модели взаимодействия «Клиент-Сервер», устройства протокола HTTP/HTTPS, жизненного цикла веб-запроса и роли серверных скриптов в формировании ответа.

Освоение инструментария: изучение синтаксиса и функциональных возможностей языка PHP, включая работу с типами данных, строковыми и массивными структурами, а также встроенными библиотеками функций.

Разработка функционала: приобретение навыков реализации ключевых механизмов веб-приложений: обработки HTML-форм, управления файловой системой, механизмами аутентификации и авторизации пользователей (сессии, куки).

Архитектурное проектирование: изучение и применение на практике современных паттернов проектирования, таких как MVC (Model-View-Controller) и Front Controller, для создания масштабируемого и поддерживаемого кода, разделения бизнес-логики и представления.

Обеспечение безопасности: изучение методов защиты веб-ресурсов от распространенных угроз (XSS, CSRF, Session Hijacking) и внедрение практик безопасной разработки (валидация, санитария данных).

Работа с данными: освоение методов хранения и обмена данными без использования реляционных СУБД (работа с текстовыми файлами, форматами JSON, XML, CSV).

Требования к результатам освоения дисциплины:

В результате освоения дисциплины обучающийся должен демонстрировать следующие результаты образования:

Знать:

– архитектуру веб-приложений и принципы работы связки «Веб-сервер (Apache/Nginx) + Интерпретатор PHP»;

– стандарты кодирования (PSR) и синтаксические конструкции языка PHP: типы данных, области видимости переменных, управляющие конструкции, механизмы подключения файлов;

– структуру и особенности протокола HTTP: назначение заголовков запроса и ответа, семантику методов (GET, POST, PUT, DELETE), коды состояния и их влияние на поведение клиента;

– механизмы управления состоянием в протоколе HTTP (Stateless): принципы работы и различия между Cookies и Sessions, настройки времени жизни и безопасности сессий;

– встроенные функции PHP для работы с файловой системой, потоками ввода-вывода, обработки строк, дат и формата JSON;

– теоретические основы шаблонизации и принципы отделения программного кода от HTML-разметки.

Уметь:

– настраивать локальную среду веб-разработки (WAMP/LAMP стек), конфигурировать файл php.ini и анализировать логи ошибок веб-сервера;

– принимать и обрабатывать данные, отправленные клиентом через HTML-формы, осуществлять их валидацию (проверку на корректность) и фильтрацию (очистку);

– реализовывать механизмы регистрации и авторизации пользователей, управлять правами доступа к закрытым разделам сайта;

– выполнять операции с файловой системой на сервере: чтение конфигураций, запись логов, реализация безопасной загрузки файлов (Upload) от пользователей;

– проектировать и реализовывать простые API-интерфейсы для обмена данными в формате JSON;

– использовать инструменты отладки и профилирования кода (xdebug, var_dump, print_r) для поиска и устранения логических ошибок.

В результате освоения учебной дисциплины обучающийся должен обладать следующими профессиональными компетенциями:

ПК 4.1. Администрировать среды и платформы разработки информационных ресурсов.

ПК 4.2. Создавать программный код на стороне сервера в соответствии с техническим заданием (спецификацией) с использованием языков программирования, библиотек и фреймворков.

ПК 4.3. Осуществлять отладку программного кода на стороне сервера на уровне программных модулей, межмодульных взаимодействий и взаимодействий с окружением.

Содержание дисциплины

Семестр № 5

1. Основы и синтаксис языка PHP

Введение в серверное программирование. Роль PHP в веб-разработке. Принципы работы связки веб-сервера и интерпретатора PHP. Базовый синтаксис языка: теги <?php ... ?>, инструкции, комментарии. Переменные, типы данных (скалярные, составные, специальные), приведение типов. Операторы: арифметические, сравнения, логические, присваивания. Конструкции для вывода данных: echo, print. Условные операторы: if, else, elseif, switch. Циклы: while, do-while, for, foreach. Основы работы с массивами: создание, индексация, ассоциативные массивы, многомерные массивы.

2. Типы запросов и пользовательские функции

Понятие HTTP-запроса и его методов. Суперглобальные массивы $_GET и $_POST. Обработка данных, отправленных через HTML-формы. Различия между методами GET и POST, их безопасность и применение. Валидация и санитация пользовательского ввода. Пользовательские функции: объявление, параметры (включая значения по умолчанию), возврат значений. Области видимости переменных: глобальная, локальная, суперглобальная. Ключевое слово global. Статические переменные внутри функций.

3. Работа с данными и состоянием приложения

Протокол HTTP как протокол без состояния (Stateless). Механизмы для поддержания состояния: Cookies и Sessions. Работа с Cookies: установка (setcookie()), чтение ($_COOKIE), удаление, параметры безопасности (expire, path, domain, secure, httponly). Работа с Sessions: старт сессии (session_start()), сохранение данных в $_SESSION, уничтожение сессии (session_destroy()).

Базовые аспекты безопасности: защита от Session Hijacking, использование сессионных токенов.

Семестр № 6

4. Расширенные возможности языка PHP

Работа с файловой системой: функции для чтения, записи, удаления файлов и директорий (fopen, fclose, fwrite, file_get_contents, file_put_contents, unlink, rmdir). Обработка ошибок ввода-вывода. Работа с форматами данных: сериализация и десериализация (serialize, unserialize). Формат JSON: кодирование (json_encode) и декодирование (json_decode), обработка ошибок JSON. Использование JSON для хранения и передачи данных в веб-приложениях.

5. Архитектура и безопасность веб-приложений

Принципы архитектуры веб-приложений. Паттерн MVC (Model-View-Controller): разделение логики, данных и представления. Реализация простого шаблонизатора для отделения HTML от PHP-кода. Понятие Front Controller и единой точки входа (index.php). Основы маршрутизации (роутинга). Безопасность веб-приложений: защита от XSS (кросс-сайтового скриптинга), CSRF (межсайтовой подделки запроса), SQL-инъекций (в контексте работы с файлами). Валидация и экранирование данных. Хеширование паролей с использованием password_hash() и password_verify().

6. Интеграция и оптимизация веб-приложений

Создание простого RESTful API для взаимодействия с клиентской частью. Обработка различных HTTP-методов (GET, POST, PUT, DELETE) в PHP. Формирование HTTP-ответов с соответствующими заголовками и кодами состояния. Буферизация вывода (ob_start, ob_get_clean). Основы кэширования на стороне сервера для повышения производительности. Профилирование и отладка PHP-кода: использование var_dump, print_r, инструментов вроде Xdebug. Логирование ошибок и событий приложения.

3 РАСПРЕДЕЛЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ПО РАЗДЕЛАМ

Цель методических рекомендаций: помочь студентам при самостоятельном изучении учебной программы с использованием лекционных материалов и рекомендуемой учебно-методической литературы.

Номер раздела данной дисциплины	Наименование тем, вопросов, вынесенных для самостоятельного изучения
Семестр № 5	

Тема 1. Изучение дополнительных возможностей массивов и функций в PHP	Рекурсивные функции и их применение. Функции для работы с массивами: array_map, array_filter, array_reduce. Создание и использование callable-типа и анонимных функций (замыканий). Практика решения алгоритмических задач на PHP.
Тема 2. Безопасная обработка пользовательского ввода и валидация данных	Подробное изучение функций фильтрации данных (filter_var, filter_input). Создание пользовательских функций валидации для сложных данных (например, email, URL, номера телефонов). Защита от распространенных уязвимостей: SQL-инъекции (при работе с файлами), XSS, CSRF. Использование токенов в формах.
Тема 3. Углубленное изучение механизмов сессий и кук	Настройка параметров сессий в php.ini (время жизни, хранение, использование cookies). Альтернативные методы хранения сессий (файлы, базы данных – обзорно). Защита от Session Fixation и Session Hijacking. Практическая реализация системы "Запомнить меня" с использованием долгоживущих кук.
Семестр № 6	
Тема 4. Работа с альтернативными форматами данных (XML, YAML).	Чтение и запись XML-файлов с использованием расширения SimpleXML или DOMDocument. Основы синтаксиса YAML. Использование библиотеки (например, Symfony YAML) для парсинга и генерации YAML в PHP. Сравнение форматов JSON, XML и YAML для хранения конфигураций и данных.
Тема 5. Изучение паттернов проектирования веб-приложений	Паттерны, применяемые в веб-разработке: Singleton, Factory, Observer, Dependency Injection (внедрение зависимостей). Принципы SOLID и их значение для написания поддерживаемого кода. Анализ и рефакторинг простого приложения с применением изученных паттернов.
Тема 6. Инструменты отладки и профилирования PHP-приложений	Настройка и использование Xdebug для пошаговой отладки. Анализ производительности кода с помощью профилировщика (например, Xdebug Profiler или Blackfire.io). Чтение и интерпретация логов веб-сервера (Apache/Nginx) и журналов ошибок PHP. Использование мониторинга для выявления узких мест в приложении.

4 КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ ДЛЯ САМОПРОВЕРКИ

1. Виды и формы самостоятельной работы студентов по дисциплине «Бэкенд-разработка»:

- изучение официальной документации языка PHP;
- выполнение практических заданий по написанию скриптов;
- реализация лабораторных работ и отладка кода;
- подготовка к зачету и экзамену.

2. Методические рекомендации:

Используйте локальный сервер (Denwer) для запуска примеров кода.

При возникновении ошибок анализируйте логи сервера (error.log) и вывод ошибок интерпретатора.

Оформите отчеты по лабораторным работам, включив листинг кода и скриншоты работы.

3. Подготовка к аттестации:

Систематизируйте знания по синтаксису и архитектуре приложений.

Убедитесь в понимании различий между клиентским (JS) и серверным (PHP) кодом.

Перечень информационных технологий

Для проведения лекционных и лабораторных занятий рекомендуется использовать программное обеспечение:

1. Debian, Simply Linux, Microsoft Windows. Системное программное обеспечение
2. LibreOffice. Программное обеспечение для работы с различными типами документов: текстами, электронными таблицами, базами данных и др.
3. Visual Studio Community. Полнофункциональная, расширяемая и бесплатная интегрированная среда разработки для создания современных приложений Android, iOS и Windows, а также веб-приложений и облачных служб
4. Denwer. Набор дистрибутивов (локальный сервер WAMP) и программная оболочка, предназначенные для создания и отладки сайтов (веб-приложений, прочего динамического содержимого интернет-страниц) на локальном ПК (без необходимости подключения к сети Интернет) под управлением ОС Windows.

Контрольные вопросы

1. В чем заключается роль серверного языка программирования?
2. Как работает связка «Веб-сервер + Интерпретатор PHP»?

3. Как объявляются переменные в PHP? Правила именования переменных.
4. Перечислите основные типы данных PHP. Является ли PHP языком со строгой типизацией?
5. В чем разница между обработкой строк в одинарных (') и двойных (") кавычках?
6. Как работают операторы сравнения == и ===?
7. Опишите синтаксис цикла foreach. Для чего он используется?
8. Что такое ассоциативный массив? Как добавить элемент в массив?
9. Какие встроенные функции для работы с массивами вы знаете (count, array_merge, in_array)?
10. Что такое область видимости переменной? Ключевое слово global.
11. Как передать данные от клиента на сервер? Разница между методами GET и POST.
12. Что такое суперглобальные массивы? Перечислите их.
13. Как получить доступ к параметрам URL (query string)?
14. Как обработать данные формы на сервере? Валидация empty, isset.
15. Что такое HTTP-заголовки? Как выполнить редиректс помощью функции header()?
16. Как прочитать содержимое файла в строку (file_get_contents)?
17. Как записать данные в файл? Режимы перезаписи и добавления (FILE_APPEND).
18. Как обработать загрузку файла на сервер (Upload)? Массив $_FILES.
19. Что такое сессия? Механизм работы (Session ID). Функции session_start, session_destroy.
20. Что такое Cookie? Как установить и удалить куку?
21. В чем отличие хранения данных в сессиях и в куках?
22. Как защитить приложение от XSS-атак? Функция htmlspecialchars.
23. Как форматировать дату и время в PHP?
24. Как подключить один PHP-файл к другому? Разница между include и require.
25. Как прочитать список файлов в директории (scandir)?
26. Что такое формат JSON? Функции json_encode и json_decode.
27. Что такое шаблонизация? Почему важно отделять HTML от PHP-логики?
28. Объясните понятие Front Controller (Единая точка входа).
29. Как реализовать простую маршрутизацию (Routing) на основе URL?
30. Как обрабатывать исключения в PHP? Конструкция try-catch.

31. Что такое CSRF-атака? Как использовать токены для защиты форм?
32. Как отправить электронное письмо из PHP-скрипта?
33. Что такое REST API? Принципы построения.
34. Как реализовать возврат данных в формате JSON для API?
35. Что такое профилирование кода? Функции отладки var_dump, print_r.
36. Опишите основные компоненты архитектуры MVC (Model-View-Controller).
37. Как происходит передача данных из Контроллера в Представление?
38. Меры безопасности при работе с пользовательскими файлами.
39. Что такое рекурсия? Как использовать её для обхода дерева папок?
40. Коды ответов HTTP (200, 301, 404, 500) и их значение.
41. Области видимости static внутри функции.
42. Анонимные функции и замыкания.
43. Понятие буферизации вывода (ob_start).
44. Основы работы с зависимостями (Composer) – обзорно.
45. Как определить, был ли запрос отправлен через AJAX?

5 ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Индивидуальное задание выполняется в течение двух семестров. Студент должен разработать динамическое веб-приложение на PHP, реализующее функционал управления контентом и пользователями.

Важное условие: Использование реляционных баз данных (MySQL, PostgreSQL) в данном курсе не предусмотрено. Все данные должны храниться в файловой системе (текстовые файлы .txt, форматы .json, .csv или .xml).

Требования к проекту:

Архитектура: Приложение должно быть построено по принципу разделения логики и представления (использование шаблонов). Рекомендуется паттерн Front Controller.

Аутентификация: Реализация регистрации и входа пользователей. Пароли должны храниться в хешированном виде (password_hash). Использование сессий.

CRUD: Реализация операций создания, чтения, редактирования и удаления данных (сущностей).

Данные: Хранение данных в файлах JSON или CSV.

Формы: Валидация всех входных данных на сервере. Защита от XSS и CSRF.

Файлы: Возможность загрузки изображений или документов на сервер (аватарок, файлов статей).

Примерный перечень предметных областей, на основании которых может быть сформирован индивидуальный вариант задания (тема веб-приложения):

1. Личный блог (Flat-file Blog): создание статей, категории, комментарии (в JSON), админка для автора.
2. Гостевая книга: форма добавления отзыва, сохранение в файл, модерация отзывов администратором, пагинация.
3. Файловый менеджер: просмотр директорий сервера, создание папок, загрузка, переименование и удаление файлов.
4. Система тестирования: конструктор тестов (вопросы в JSON), прохождение теста студентом, расчет результата, сохранение истории.
5. Фотогалерея: массовая загрузка фото, автоматическое создание превью (если позволяет библиотека GD), альбомы, описание фото.
6. Интернет-магазин (без БД): каталог товаров из CSV, корзина (в сессии), оформление заказа (сохранение заказа в файл), админка товаров.
7. Система голосования: создание опросов, варианты ответов, сохранение голосов, защита от накрутки (по IP/Cookie), вывод результатов.
8. Простой форум: создание тем (папки), сообщений (файлы), профили пользователей, права модераторов.
9. Контактная книга (CRM): карточки клиентов, поиск, фильтрация, экспорт базы контактов в CSV.
10. Генератор статических сайтов: форма ввода контента, выбор шаблона, генерация готовых HTML-страниц в папку.
11. Сервис коротких ссылок: админка для создания ссылок, хранение пар "код-URL" в JSON, редирект, статистика переходов.
12. Органайзер задач (ToDo): личные списки задач для пользователей, статусы, дедлайны, сохранение в личный JSON-файл пользователя.
13. Сайт рецептов: добавление рецептов, поиск по ингредиентам, загрузка фото блюда, избранное.
14. Система техподдержки (Тикеты): создание заявки пользователем, ответы администратора, смена статуса заявки.
15. Электронный дневник: список студентов (CSV), предметы, выставление оценок, расчет среднего балла.
16. Агрегатор новостей: парсинг (чтение) RSS-лент других сайтов, сохранение кэша в файлы, вывод единой лентой.
17. Викторина "Кто хочет стать миллионером": база вопросов в файле, логика подсказок, таблица рекордов.
18. Сервис для проведения аукционов: лоты (JSON), ставки пользователей, таймер (логика на сервере), определение победителя.

19. База знаний (Wiki): создание страниц, редактирование контента (Markdown или HTML), история изменений (версии файлов).
20. Анкетирование: конструктор анкет, сбор ответов в CSV-файл, выгрузка результатов в Excel.
21. Календарь событий: добавление событий на дату, просмотр расписания на месяц, напоминания (визуальные).
22. Система управления недвижимостью: объявления о сдаче/продаже, фото, фильтры, контакты риелтора.
23. Библиотека документов: загрузка doc/pdf файлов, категоризация, поиск по описанию, учет скачиваний.
24. Сервис "Тайный Санта": регистрация участников, автоматическое распределение (перемешивание массива), отправка результатов (имитация email).
25. Чат (Simple Chat): сохранение сообщений в общий лог-файл, AJAX-обновление (long polling или просто таймер), список онлайн.
26. Калькулятор доставки: админка тарифов (JSON), форма расчета для клиента, сохранение заявки на доставку.
27. Портфолио фрилансера: админка для добавления проектов, загрузка скриншотов, редактирование описания "О себе".
28. Система бронирования переговорных: сетка времени, занятие слота пользователем, отмена брони.
29. Сервис обмена заметками (Pastebin): создание текстовой заметки, генерация уникальной ссылки, удаление по таймеру или паролю.
30. Учет домашних финансов: доходы/расходы, категории, баланс, формирование отчета за месяц в HTML.

Примерный план выполнения индивидуальной работы

31. Выбор темы. Согласование функционала с преподавателем.
32. Проектирование данных. Описание структуры JSON-объекта, который будет описывать сущность (например, Задача: id, текст, дата, статус).
33. HTML-верстка. Создание каркаса приложения (Header, Main, Footer, контейнеры для списков).
34. Стилизация (CSS). Оформление интерфейса, адаптивность.
35. Разработка Модели (JS). Написание функций для работы с данными (массивом объектов): добавление, поиск, удаление.
36. Разработка Представления (JS). Написание функций отрисовки (render), которые превращают данные в HTML-код.
37. Разработка Контроллера (JS). Назначение обработчиков событий на кнопки, формы, поля ввода.
38. Интеграция с LocalStorage. Добавление сохранения данных при любом изменении.

39. Отладка. Тестирование всех сценариев использования, проверка консоли на наличие ошибок.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Тузовский, А. Ф. Проектирование и разработка web-приложений : учебник для среднего профессионального образования / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2025. — 219 с. — (Профессиональное образование). — ISBN 978-5-534-16767-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/565693
2. Сысолетин, Е. Г. Разработка интернет-приложений : учебник для среднего профессионального образования / Е. Г. Сысолетин, С. Д. Ростунцев. — Москва : Издательство Юрайт, 2025. — 80 с. — (Профессиональное образование). — ISBN 978-5-534-19603-0. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/565692
3. Полуэктова, Н. Р. Разработка веб-приложений : учебник для среднего профессионального образования / Н. Р. Полуэктова. — 2-е изд. — Москва : Издательство Юрайт, 2025. — 204 с. — (Профессиональное образование). — ISBN 978-5-534-18644-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/567621
4. Никсон, Р. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Робин Никсон. – 5-е изд. – Санкт-Петербург : Питер, 2021. – 816 с.
5. Котеров, Д. В. PHP 7. В подлиннике / Д. В. Котеров, И. В. Симдянов. – Санкт-Петербург : БХВ-Петербург, 2019. – 1088 с.
6. Скляр, Д. PHP. Рецепты программирования / Дэвид Скляр, Адам Трахтенберг. – 3-е изд. – Москва : Эксмо, 2017. – 784 с.
7. Веллинг, Л. Разработка веб-приложений с помощью PHP и MySQL / Люк Веллинг, Лаура Томсон. – 5-е изд. – Москва : Вильямс, 2019. – 1072 с.
8. Локхарт, Д. Современный PHP: новые возможности и передовой опыт / Джош Локхарт. – Москва : ДМК Пресс, 2016. – 304 с.
9. Зандстра, М. PHP: объекты, шаблоны и методики программирования / Мэтт Зандстра. – 5-е изд. – Москва : Вильямс, 2019. – 576 с.
10. Официальная документация PHP [Электронный ресурс]. – Режим доступа: https://www.php.net/manual/ru/ – Загл. с экрана.
11. PHP: The Right Way (PHP: Правильный путь) [Электронный ресурс]. – Режим доступа: https://phptherightway.com/ – Загл. с экрана.
12. Стандарты PSR (PHP Standards Recommendations) [Электронный ресурс] // PHP-FIG. – Режим доступа: https://www.php-fig.org/psr/ – Загл. с экрана.