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

Панасов В.Л.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ЛАБОРАТОРНЫХ РАБОТ И САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ ПО ДИСЦИПЛИНЕ

МДК.02.01 «АДМИНИСТРИРОВАНИЕ ИНФОРМАЦИОННЫХ РЕСУРСОВ»

для специальности
09.02.09 Веб-разработка

Ростов-на-Дону
2025

СОДЕРЖАНИЕ

Введение

Методические указания к лабораторным работам и по выполнению самостоятельной работы студентов составлены в соответствии с ФГОС СПО и рабочей программой профессионального модуля МДК.02.01 «Администрирование информационных ресурсов», которые являются частью программы подготовки специалистов среднего звена специальности 09.02.09 Веб-разработка.

Рабочей программой дисциплины МДК.02.01 «Администрирование информационных ресурсов» предусмотрено на выполнение лабораторных работ – 28 часов и самостоятельной работы студентов – 50 часов.

При выполнении лабораторных работ и при организации самостоятельной работы студентов используются активные и интерактивные формы обучения - просмотр и обсуждение учебных видеофильмов, групповая дискуссия, лекция - консультация, моделирование производственных процессов и ситуаций, обсуждение в группах, тренинг, кейс-метод, защита практических и лабораторных работ и другие.

Цель методических рекомендаций - оказание методической помощи студентам в выполнении лабораторных работ и в организации их самостоятельной работы по изучению учебного материала, для расширения, углубления и закрепления знаний и умений, а также формирования профессиональных (ПК) компетенций.

Код и содержание компетенции	Уметь	Знать
ПК 2.1 - Устанавливать прикладное программное обеспечение и модулей информационных ресурсов, включая их настройку	соблюдать процедуру установки прикладного программного обеспечения в соответствии с документацией; идентифицировать инциденты, возникающие при установке программного обеспечения, и принимать решение по изменению процедуры установки; пользоваться нормативно-	принципы устройства и функционирования информационных ресурсов; принципы устройства и функционирования программных средств и платформ для разработки веб-ресурсов;

	технической документацией в области программного обеспечения; производить настройку параметров веб-сервера; устанавливать систему управления базами данных (СУБД);	
ПК 2.2 - Проводить работы по резервному копированию и развертыванию резервной копии информационных ресурсов	выполнять регламентные процедуры по резервированию данных; устанавливать прикладное программное обеспечение для резервирования информационных ресурсов.	основы управления изменениями; основы резервного развертывания и резервного копирования информационных ресурсов; общие основы решения практических задач по созданию резервных копий; возможности ИР.
ПК 2.3 - Настраивать права пользователей в соответствии с функциональными задачами (ролями) и на основании информации о поведенческих факторах.	пользоваться нормативно-технической документацией в области программного обеспечения; идентифицировать права пользователей в зависимости от функционала информационного ресурса; регламентировать уровни прав и ролей пользователей информационных ресурсов; применять регламентные	принципы устройства и функционирования информационных ресурсов; современные стандарты взаимодействия компонентов распределенных приложений; возможности ИР.

	процедуры управления правами доступа пользователей информационных ресурсов.	
ПК 2.4 - Применять программные средства обеспечения безопасности информации веб приложений	пользоваться нормативно-технической документацией в области программного обеспечения; производить настройку параметров веб-сервера;	принципы устройства и функционирования информационных ресурсов; программные средства и платформы для разработки веб-ресурсов; основы информационной безопасности веб-ресурсов; современные стандарты взаимодействия компонентов распределенных приложений; принципы использования электронно-цифровых подписей и работы удостоверяющих центров;
ПК 2.5 - Обрабатывать запросы заказчика в службе технической поддержке в соответствии с трудовым заданием	выяснять из беседы с заказчиком и понимать причины возникших аварийных ситуаций с информационным ресурсом; применять установленные правила делового общения при общении с заказчиком; отвечать на запросы заказчика в установленные регламентом сроки; анализировать и решать типовые запросы заказчиков;	принципы устройства и функционирования информационных ресурсов; основы управления изменениями; возможности ИР; инструменты и методы коммуникаций; каналы коммуникаций; модели коммуникаций; технологии межличностной и групповой коммуникации в деловом взаимодействии, основы конфликтологии.

	работать с программным обеспечением по приему, обработке и регистрации запросов заказчика; координировать решение запросов заказчиков со специалистами соответствующих подразделений; объяснять заказчикам пути решения возникшей проблемы.	

1 ПЛАН РАСПРЕДЕЛЕНИЯ УЧЕБНОЙ НАГРУЗКИ

Объем дисциплины в академических часах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Вид учебной работы	**Объем часов**
Объем образовательной программы учебной дисциплины	**108**
в том числе:	
Лекции (теоретическое обучение)	28
Лабораторные работы	28
Самостоятельная работа	50
Промежуточная аттестация (в форме зачета)	**2**

Содержание дисциплины

Наименование лекционных занятий	**Трудоемкость аудиторной работы, часы**
Раздел № 1 Установка прикладного программного обеспечения и модулей информационных ресурсов	
1.1 Основы работы в операционных системы Linux и Windows.	2
1.2 Описание протокола DNS и HTTP/HTTPS, особенности функционирования веб-приложения	2
1.3 Тонкости установки и настройки LAMP и WAMP или аналогов.	2
1.4 Особенности развертывания готовых систем CMS, LMS, CRM и установки дополнений. Виды хостингов и особенности их использования.	2
Раздел № 2 Обработка запросов заказчика в службе технической поддержки	
2.1 Технологии межличностной и групповой коммуникации в деловом взаимодействии, основы конфликтологии.	2

Наименование лекционных занятий	Трудоемкость аудиторной работы, часы
2.2 Инструменты, каналы, модели, методы коммуникации.	2
2.3 Основы управления изменениями.	2
2.4 Методология управления, отладки и непрерывного улучшения бизнес-процессов, связанных с ИТ на примере ITIL. Принципы устройства и работы служб технической поддержки.	
Раздел № 3 Резервное копирование информационных ресурсов	
3.1 Понятие безопасности данных. Особенности работы с хостингами и выделенными серверами.	2
3.2 Основы резервного копирование и восстановления. Особенности работы с файловой системой.	2
3.3 Особенности работы с базой данных.	2
Раздел № 4 Программные средства обеспечения безопасности информации веб-приложений и настройка ролей пользователей	
4.1 Основы информационной безопасности веб-ресурсов. Принципы использования электронно-цифровых подписей и работы удостоверяющих центров. Способы написания безопасного программного кода.	2
4.2 Программные средства обеспечения безопасности функционирования веб-приложений. Виды организации контроля доступа к системам и способы распределения прав.	2
4.3 Регламентирование и учет доступа к системам. Внутренние и внешние технические способы обеспечения контроля прав пользователей, в том числе распределенные.	2

Лабораторные работы

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы

Наименование (тематика) лабораторных работ, семинаров	**Трудоемкость аудиторной работы, часы**
Раздел № 1	
1.1 Развертывание операционной системы.	2
1.2 Установка и настройка WAMP подобного комплекта.	2
1.3 Установка и настройка готовой CMS WordPress.	2
1.4 Установка системы функционирования технической поддержки Hesk.	2
1.5 Установка среды веб-разработки на базе VS Code.	2
1.6 Публикация веб-приложения на хостингах разного типа.	2
Раздел № 2	
2.1 Составление блок-схемы работы оператора технической поддержки.	2
2.2 Выполнение обработки запросов в специализированной информационной системе.	2
2.3 Решение и разбор примеров критических ситуаций в службе поддержки.	1
Раздел № 3	
3.1 Резервное копирование и восстановление файловой системы веб-браузера.	2
3.2 Резервное копирование и восстановление базы данных веб-приложения.	2
3.3 Использование сценариев и скриптов для организации процесса резервирования и восстановления данных.	2
Раздел № 4	
4.1 Настройка прав доступа к файловой системе и базе данных.	1
4.2 Настройка ролей доступа пользователей в CMS.	1

Наименование (тематика) лабораторных работ, семинаров	Трудоемкость аудиторной работы, часы
4.3 Анализ безопасности веб-сервиса на предмет наличия уязвимостей.	1
4.4 Настройка веб-сервера с использованием протокола HTTPS.	1
4.5 Настройка программного файрволла для веб-приложения.	1

Самостоятельное изучение учебного материала (самоподготовка)

Номер раздела данной дисциплины	Наименование тем, вопросов, вынесенных для самостоятельного изучения	Трудоемкость внеаудиторной работы, часы
Семестр № 5		
1	Установка ОС Ubuntu.	10
1	Установка и настройка LMS Moodle.	10
2	Системы HelpDesk и ServiceDesk.	10
3	Резервное копирование БД PostgreSQL и MS SQL.	10
4	Виды атак на Web-серверы и методов борьбы с ними.	10

Лабораторная работа №1 «Развертывание операционной системы»

Теоретические сведения

Рассмотрим, как автоматически развертывать Windows 10 по сети с помощью SCCM (System Center Configuration Manager) и PXE. С помощью данной методики вы можете существенно сократить время на установку ОС рабочих станций и использовать для массового развертывания Windows 10 на новых компьютерах.

Одна из функции SCCM — возможность сетевой установки операционной системы Windows системы на большое количество компьютеров вместе с драйверами, программами, обновлениями. Предполагаем, что у вас уже имеется настроенная инфраструктура SCCM.

Настройка PXE роли на SCCM сервере для сетевой установки Windows.

Функционал PXE (Preboot Execution Environment) в современных компьютерах позволяет выполнить сетевую установку операционной системы без применения каких-либо носителей. В SCCM его настройка происходит в окне свойств роли точки распространения (в контекстном меню вам нужно выбрать пункт Properties) и установить флажки:

- Enable PXE support for client;
- Allow this distribution point to respond to incoming PXE request;
- Enable unknown computer support;
- Require a password when computers use PXE.

Также желательно выбрать из выпадающего списка Allow user device affinity with automatic approval.

После включения этих настроек на сервере будет установлена роль Windows Deployment Services, интегрированная с SCCM (рис. 1).

Создание настроенного эталонного образа Windows 10.

Следующий этап – подготовка эталонного образа с Windows 10, который будет устанавливаться на другие компьютеры. Проще всего взять типовой компьютер, установить на него Windows 10. Установите последние обновления безопасности, отключите ненужные службы, удалите встроенные приложения, установите драйвера. Настройте Windows 10 в соответствии со своими требованиями. Это компьютер не нужно вводить в домен AD.

Создаем загрузочный образ для захвата эталонного образа Windows 10.

По умолчанию SCCM содержит образы Windows PE (среда предустановки Windows, это загрузочные файлы Boot image x86.wim и Boot image x64.wim), которые необходимы для настройки операционной системы во время “захвата”.

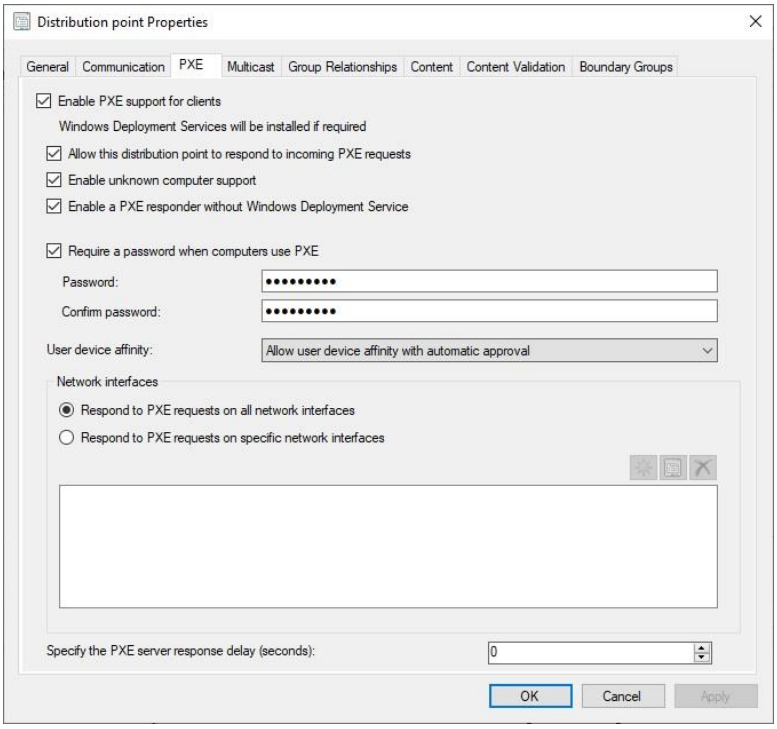


Рис. 1. Настройка PXE

Сначала нужно добавить образ Boot image x64.wim в вашу точку распространения SCCM. Для этого зайдите в *Software Library-> Operation System-> Boot Images*, в контекстном меню выберите пункт *Distribute Content*, затем точку распространения и *OK*. Цветовой окрас состояния сменится с серого на жёлтый, а по окончанию – на зелёный (Рис. 2).

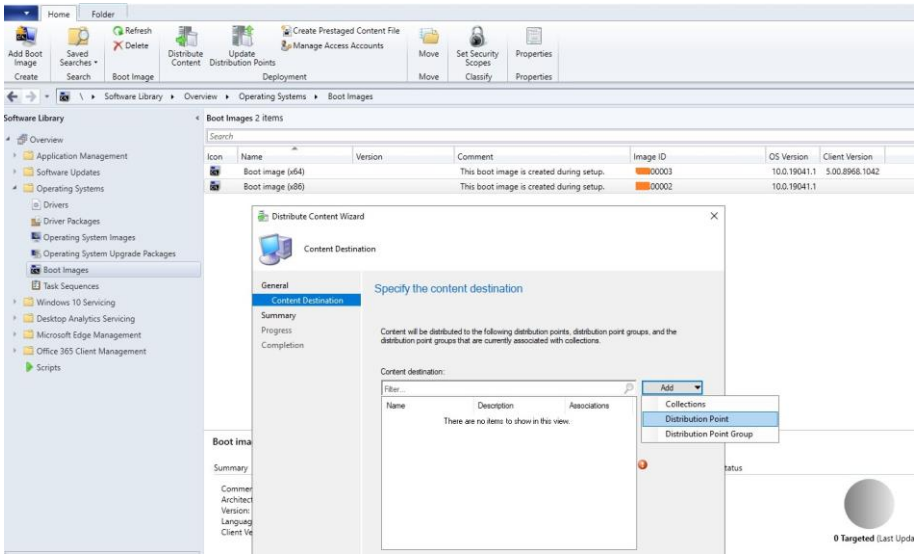


Рис. 2. Добавление образа

Теперь нужно создать иметь загрузочные файлы для “захвата” эталонной операционной системы по сети. Для их создания перейдите в *Software Library-> Operation System -> Task Sequences* и в контекстном меню выберите *Create Task Sequence Media*. В появившемся окне предлагается 4 варианта создания образа (рис. 3):

- *Stand-alone media* — создание автономного образа для установки ОС (локальная установка, без использования сетевой загрузки);
- *Bootable media* – создание загрузочного образа, для распространения которого используя инфраструктура центра конфигураций;
- *Capture media* – создание загрузочного образа для захвата эталонного образа компьютера;
- *Prestaged media* – создание предварительного образа для нового жёсткого диска, который включает образ операционной системы. Используется для старых карт, не поддерживающих PXE загрузку.

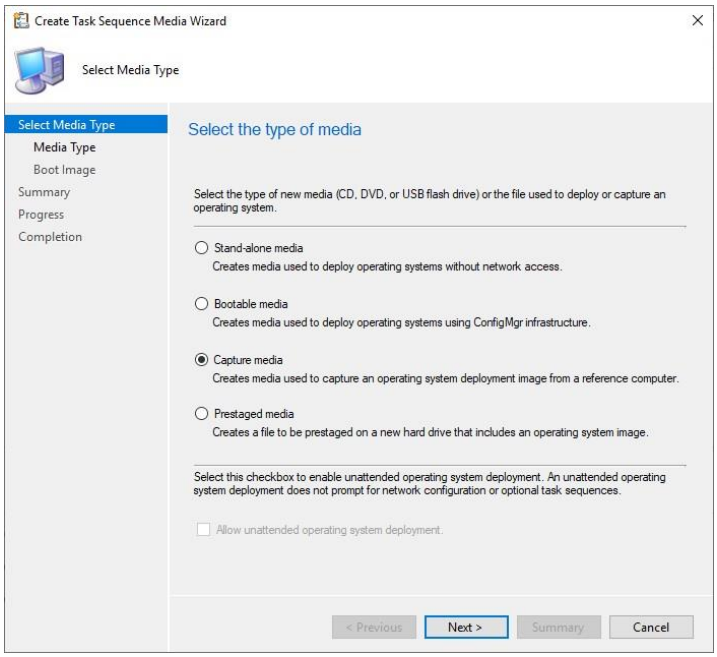


Рис. 3. Варианты образов

Выберите пункт *Capture media*, следуйте подсказкам мастера установки. Укажите место сохранения iso-файла, выберите Boot image x64.wim и точку распространения, *Next->Next->Close*. Процесс длится около 1 минуты.

Далее созданный полученный образ для захвата ОС нужно подключить и запустить на подготовленном ПК (файл LaunchMedia.cmd). Укажите место сохранения полученного образа Windows 10.

Весь процесс захвата длится примерно 40 минут. По окончании получаете эталонный wim-образ размером примерно 4.5 Гб, который нужно

скопировать на сервер SCCM. Далее вам необходимо его добавить в точку распространения. Для этого заходите в *Software Library -> Operation Systems -> Operation System Images*, в контекстном меню выберите пункт *Add Operation System Images*, в мастере настроек укажите UNC путь к месту хранения созданного wim-файла с эталонным образом.

В контекстном меню выберите *Distribute Content* для копирования образа на вашу SCCM Distribution Point.

Создаем последовательность задач (Task Sequence) для сетевой установки Windows.

Последовательность задач SCCM (task sequence) обеспечивает пошаговое выполнение команд и действий по установке ОС, ПО, драйверов, обновлений. Для её создания зайдите в *Software Library -> Operation System -> Task Sequences* и в контекстном меню выберите *Create Task Sequence*. В появившемся окне мастер предлагает несколько вариантов:

- *Install an existing image package* –создание очереди задач для имеющегося wim образа;
- *Build and capture a reference operating system image* –захват эталонного образа;
- *Create a new custom task sequence* – создание нового пустого задания (настройка выполняется вручную). Отсутствует очередь задач по умолчанию.

Так как эталонный образ wim образ у вас уже есть, выберите первый пункт. Далее мастер предлагает поэтапно произвести большое количество настроек, но можете указать только минимально необходимые параметры (рис. 4):

1. Укажите название задачи и выберите загрузочный PXE образ Boot image x64.wim;
2. Выберите созданный ранее эталонный образ Windows 10;
3. Установите пароль локального администратора (можно автоматически сменить пароль после добавления компьютера в домен с помощью LAPS);
4. Настройте параметры автоматического присоединения в домен AD, выбрав домен и OU, в которую нужно поместить новый компьютер;
5. Выберите пользователя, под которым будет осуществляться добавление в домен;

6. На шаге *State Migration* убрать все флажки (*Capture user settings and files, Capture network settings, Capture Microsoft Windows settings*).

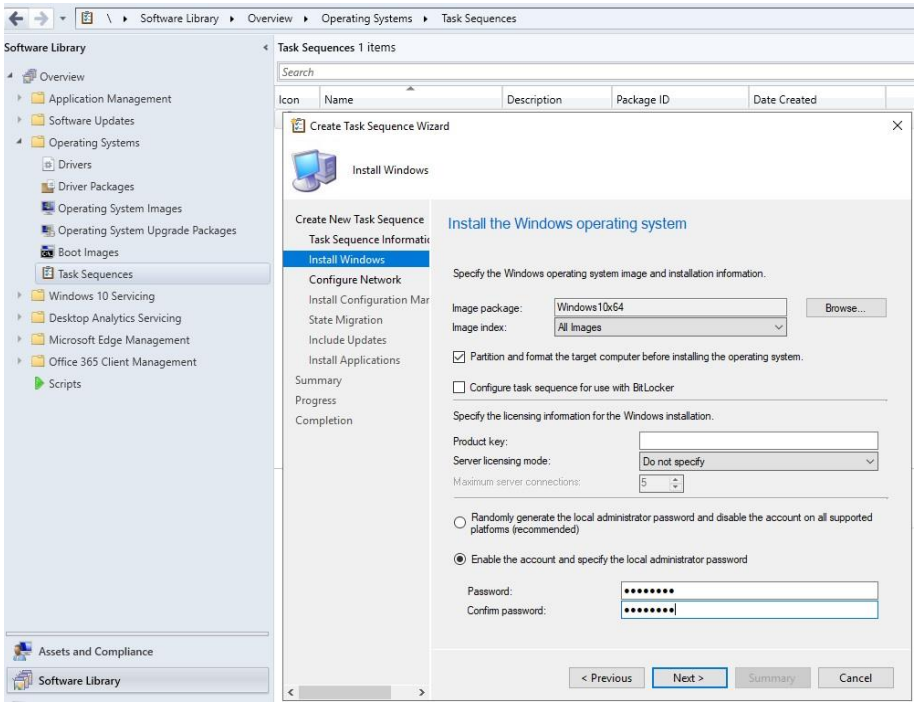


Рис. 4. Параметры образа

Последовательность задач (рис. 5) создана, но она требует от вас корректировки и проверки. Перейдите в режим редактирования Task Sequence, выбрав в контекстном меню пункт *Edit*.

В дереве справа указан список действий, которые автоматически будут выполнены с компьютером при установке Windows 10 по сети. На 2 и 3 шаге указываются параметры разбиения дисков на разделы. На шаге 2 вы можете удалить все разделы жёсткого диска, созданные по умолчанию, если вы их не используете для восстановления Windows. Далее создаёте новый раздел, называете “C:”, выбираете тип *Primary*, устанавливаете фиксированный размер 50 Гб (или другой объём), выбрав опцию *Use specific size*. Выберите что диск нужно отформатировать в файловой системе *NTFS (*флажок *Quick format)*.

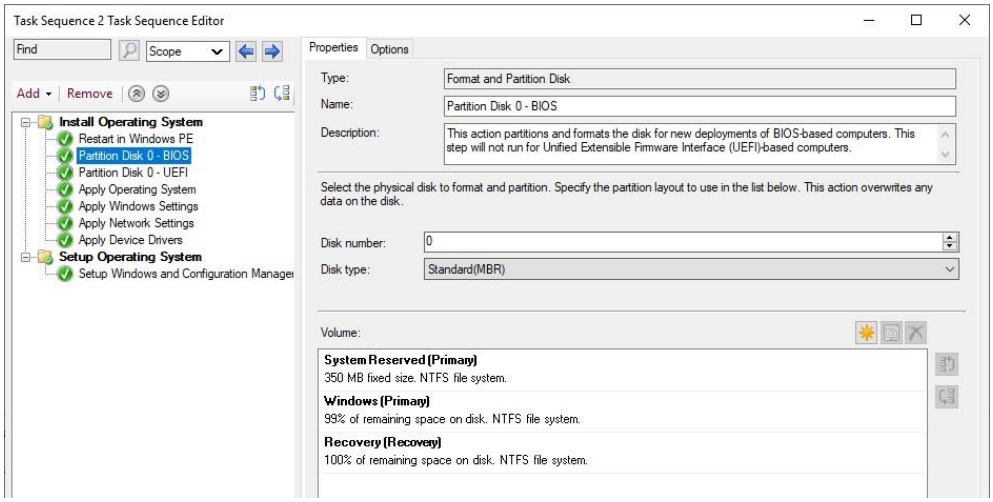


Рис. 5. Последовательность задач

Далее создаёте второй диск, называете “D”, выбираете тип *Primary*. Чтобы он занял все оставшееся неразмеченное пространство выберите опцию *Use a percentage of remaining free space*, выберите значение *100*, файловую систему *NTFS* и флажок *Quick format*.

Таким образом у вас будет создан диск C: с фиксированным размером, а всё оставшееся место уйдёт диску D: (если объём жёсткого диска большой, то можно разбить его на большее количество разделов), которые будут отформатированы автоматически.

Третий шаг в нашем примере удаляется, т.к. у нас компьютеры с BIOS. Для компьютеров с UEFI – нужно настраивать дисковые разделы именно в 3 пункте.

На следующем шаге – *Apply Operating System* – нужно выбрать раздел для установки операционной системы. Проверяете наличие эталонного образа, а внизу окна выбираете установку Windows на:

- Specific disk and partition — конкретный номер диска и номер раздела;
- Specific logical drive letter – указанный логический диск;
- Logical drive letter stored in a variable —

Например, будет второй пункт и диск C: .

Остальные шаги проверяете на отсутствие ошибок. На шаге *Apply Windows Settings* нужно указать лицензионный ключ (можно указать ключ KMS активации), пароль администратора и часовой пояс. Корректировка доменных (сетевых) настроек происходит в *Apply Network Settings*, а *Apply Device Drivers* позволяет добавить драйвера в установку.

Далее новое задание нужно опубликовать. Для этого в контекстном меню выберите пункт *Deploy*, на первом шаге мастера выбираете коллекцию

устройств *All Unknown Computers*, на 2м шаге параметру *Purpose* присваиваете значение *Available* (*Available* – доступно для выбора, *Required* – принудительная установка), в параметре *Make available to the following* выберите *Configuration manager clients, media and PXE*, остальные шаги можно оставить по умолчанию.

На этом основные действия по подготовке эталонного образа и настройке SCCM завершены.

Добавление драйверов в загрузочный образ Windows.

Рассмотренных выше действий будет достаточно для автоматической установки Windows 10 на новые современные миниатюрные системные блоки или моноблоки. В случае с классическими системниками и внешними сетевыми картами загрузка по сети может не заработать. В этом случае необходимо в загрузочный образ добавить драйвера для используемых моделей сетевых адаперов.

Для добавления драйвера в базу сайта SCCM перейдите в *Software Library -> Operation Systems -> Drivers* и в контекстном меню выберите *Import Driver*. Далее мастер поможет заполнить необходимые поля. Нужно UNC путь к каталогу с драйвером и установить для параметра *Specify the option for duplicate drivers* значение *Do not import the driver*.

Затем нужно зайти в свойства загрузочного образа Boot image x64.wim и на вкладке *Drivers* добавить драйвера.

Настройка PXE на компьютерах.

На компьютерах, которые вы хотите деплоить по сети нужно включить в BIOS поддержку сетевой загрузки. У разных производителей материнских плат такие настройки будут находится в разных местах и называться по-разному. Скорее всего в названии параметра сетевой загрузки должно быть что-то вроде PXE Boot, Network Boot, Network Card.

Например, активация PXE в UEFI моделях HP Pro 3520 и HP Pro 6300 происходит в разделе *Security -> Network Boot* установкой параметра *Enable*.

Теперь для старта всё готово. При начальной загрузке ПК происходит определение его параметров (POST). Затем DHCP сервер назначает IP-адрес сетевой карте. Для начала сетевой установки ОС в течение 5 секунд нужно нажать клавишу *F12* (означает PXE-подключение), после чего SCCM копирует на ПК загрузочные файлы (в моделях HP нужно нажимать на *F9*, возможно дополнительно ещё выбрать пункт *Network Controller* для загрузки по сети).

Далее появляется окно для выбора задачи, после чего начинается поэтапное развёртывание операционной системы, то есть выполнение тех действий, которые указаны в вашем Task Sequence. Ход установки

отслеживается при помощи индикатора. Примерно за 20 минут ваш эталонный образ Windows 10 будет установлен на новый ПК и компьютер введен в домен. Если одновременно запустить сетевую установка Windows с SCCM на 10-20 компьютерах, время установки может немного увеличиться.

По окончанию установки ОС компьютеру по умолчанию присваивается имя MININT-<7символов>, например, MININT-5EFG9DR. Это можно увидеть в консоли SCCM в разделе *Assets and Compliance -> Devices*. Вы можете изменить имя компьютера на более запоминающееся, например, BUH-OLGA. Для этого нужно зайти в Свойства компьютера -> *Изменить параметры -> Изменить,* указать новое имя компьютера и перезагрузить ПК. Вскоре изменения отобразятся как в DNS, так и в SCCM.

Установка приложений на новый компьютер с помощью SCCM.

Следующие этап установка на новый компьютер необходимых приложений. Программы можно устанавливать в виде пакета (Package) или как приложение (Application). Приложения – программы, имеющие установочные файлы с расширениями msi, appx, xap, ipa, apk и прочее. У Application возможностей больше, плюс пользователи тоже могут принимать участие в работе. Package – это специальный контейнер (обёртка), который может включать в себя несколько файлов (дистрибутивов). Используются exe-, vbs-, cmd-, cab-файлы и другие. Он проще в настройках.

После создания пакетов установки программ, вы можете добавить из в последовательность задачи развертывания Windows 10 (task sequence).

Здесь создаёте группу Software Install, в меню *Add -> Software -> Install Package* добавьте созданные пакеты программ.

Последовательность выполнения лабораторной работы

1. Узнать у преподавателя конфигурацию SCCM.

2. Настроить и запустить виртуальную машину Hyper-V.

3. Подготовить образ Windows, используя теоретические сведения выше.

4. Развернуть образ на виртуальной машине.

5. Показать результат преподавателю.

6. Ответить на вопросы преподавателя.

Лабораторная работа №2 «Установка и настройка WAMP подобного комплекта»

Теоретические сведения

WAMP-сервер является реализацией классического LAMP-сервера, но для ОС семейства Windows. В данный дистрибутив входит Apache, PHP, MySQL и автоматический установщик расширений.

Перед установкой WAMP-сервера установим недостающие библиотеки:

- Visual C++ Redistributable for Visual Studio 2015.
- Microsoft Visual C++ 2008 SP1 Redistributable Package (x64)
- Microsoft Visual C++ 2010 SP1 Redistributable Package (x64)
- Visual C++ Redistributable for Visual Studio 2012 Update 4
- Update for visual C++ 2013 Redistributable Package

Переходим на сайт проекта WAMP. Открываем раздел “Загрузка” (рис. 6)

Рис. 6. Страница загрузки WAMP

В самом начале, программа предлагает выбрать язык для всего процесса установки. Выбираем по удобству. В нашем случае будет английский. Кликаем по кнопке “OK” (рис. 7).

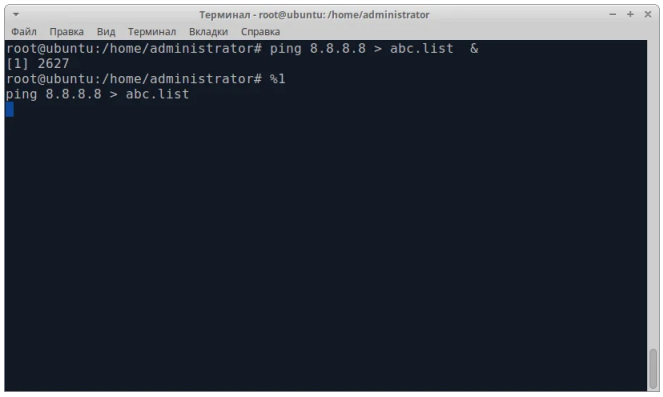


Рис. 7. Начальный запуск установки WAMP

Внимательно читаем лицензионное соглашение. Если все устраивает - принимаем условия и продолжаем установку.

На следующем этапе программа предложит ознакомиться с информацией по установке (рис. 8).

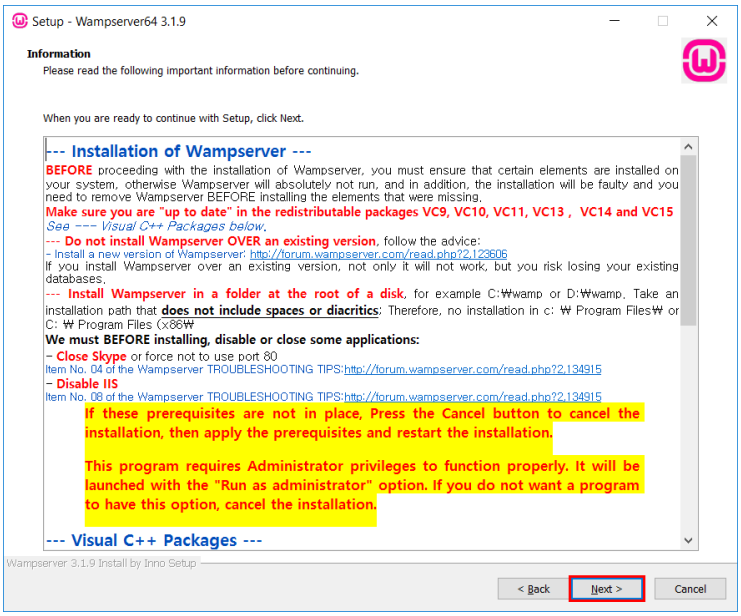


Рис. 8. Информация об установке WAMP

Данная информация напоминает о необходимости отключить IIS сервер и обновить пакеты Visual C++.

Выбираем путь установки. По умолчанию “C:wamp”. Важно, чтобы на диске было свободно более 2,5Гб. Если все устраивает, нажимаем “Далее” (Next).

На следующем шаге, программа предложит создать ярлык в меню Пуск. Кликаем “Далее” (Next).

Проверяем параметры установки и кликаем “Установить” (Install).

В процессе установки может появиться вопрос об использовании Internet Explorer в качестве браузера WAMP-сервера. Если желаете выбрать другой браузер, нажимаем кнопку “Да”. В этом случае будет необходимо указать exe-файл нового браузера.

Если установлены все вышеперечисленные обновления, сервер WAMP-запустится в системном трее (рис. 9).

Рис. 9. Значек запущенного WAMP

Проверяем работу сервера. Для этого перейдем по ссылке http://127.0.0.1.

Настройка Apache.

Разрешим подключение для всех, а не только для локальных пользователей. Отредактируем файл C:wamp64binapacheapache2.4.39confhttpd.conf.

В данном конфигурационном файле необходимо отыскать секцию:

```
<Directory "${INSTALL_DIR}/www/">
```

Заменяем значение “Require local”на “Require all granted” (рис. 10).

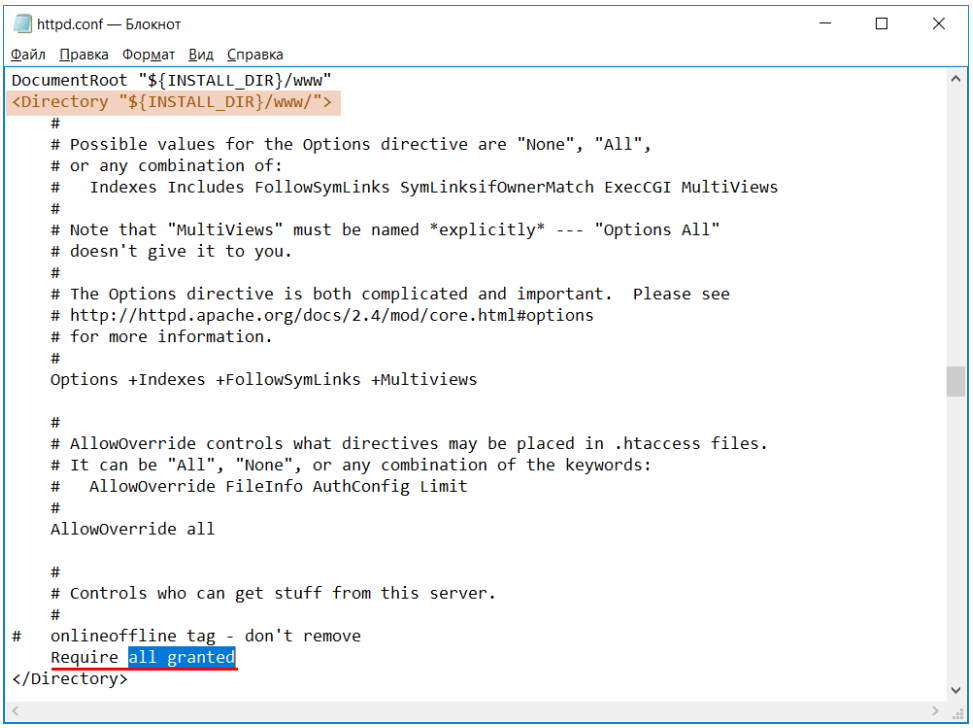


Рис. 10. Настройка прав в httpd.conf

Сохраняем изменения и закрываем файл.

Теперь отредактируем файл C:wamp64binapacheapache2.4.39confextrahttpd-vhosts.conf. Заменяем параметры и значения секции Directory на строчки, показанные на рис. 11.

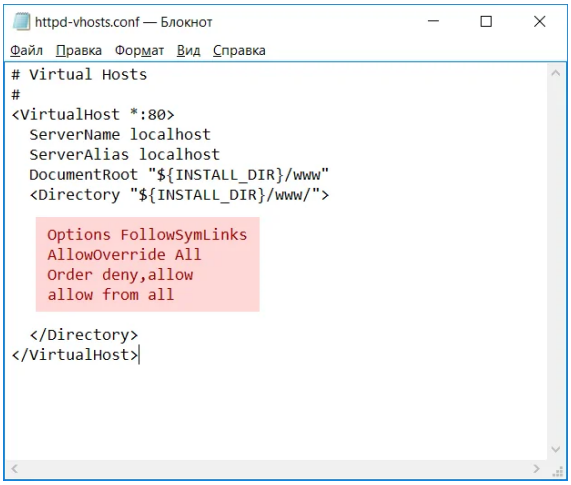


Рис. 11. Настройка виртуального хоста

Сохраняем и закрываем файл.

Перезапускаем WAMP-сервер. Для этого кликаем по значку в трее правой кнопкой мыши и выбираем Refresh (рис. 12).

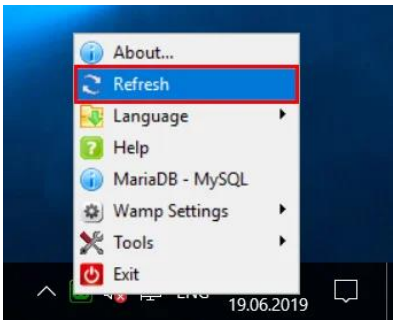


Рис. 12. Перезапуск WAMP

Для проверки работы www-сервера, создадим html-файл в корне сайта. Для этого открываем Блокнот и наполняем его html-кодом. Затем сохраним файл по пути C:wamp64wwwindex.html (рис. 13).

Теперь с другого компьютера откроем файл. Это можно сделать по доменному имени, если оно приобретено и настроено, либо по IP-адресу. В качестве примера:

`http://<domain_name>/index.html`

`http://<IP-address>/index.html`

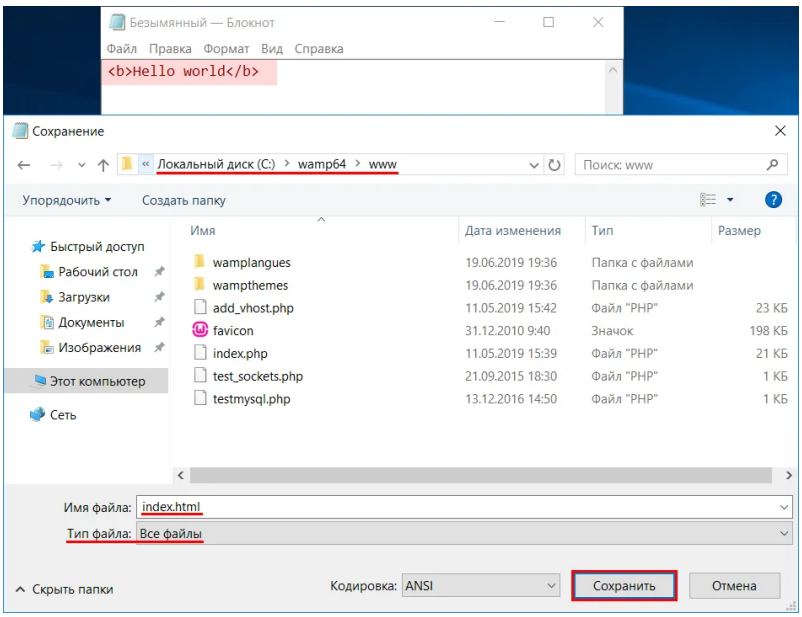


Рис. 13. Создание тестовой html-страницы

Для того, чтобы узнать свой IP-адрес, достаточно открыть командную строку или окно PowerShell и выполнить команду:

```
ipconfig
```

Открываем и видим следующую страницу (рис. 14).

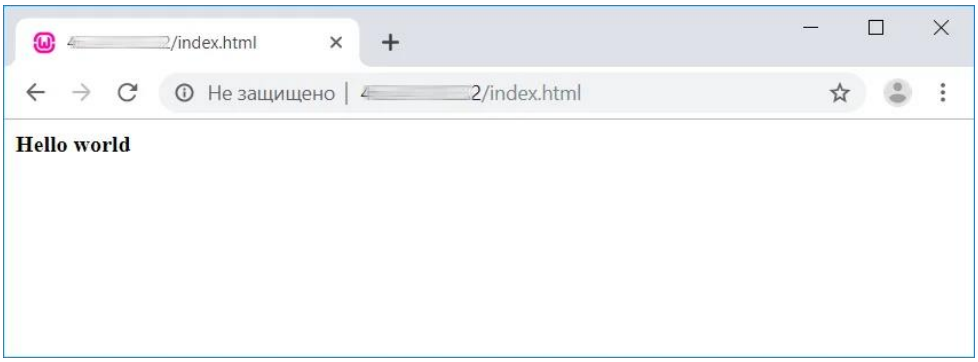


Рис. 14. Проверка тестовой html-страницы

Обратите внимание, что правила Брандмауэра Windows должны разрешать подключение к 80 и 443 TCP-портам.

Настройка phpMyAdmin.

На главной странице сервера, в самом низу, слева, в секции Tools кликаем по phpmyadmin (рис. 15).

Страница откроется по адресу http://127.0.0.1/phpmyadmin/.

Web-приложение запросит логин и пароль. По умолчанию, логин - root, пароль следует оставить пустым. Также следует выбрать используемую СУБД и кликнуть по кнопке “Вперед”.

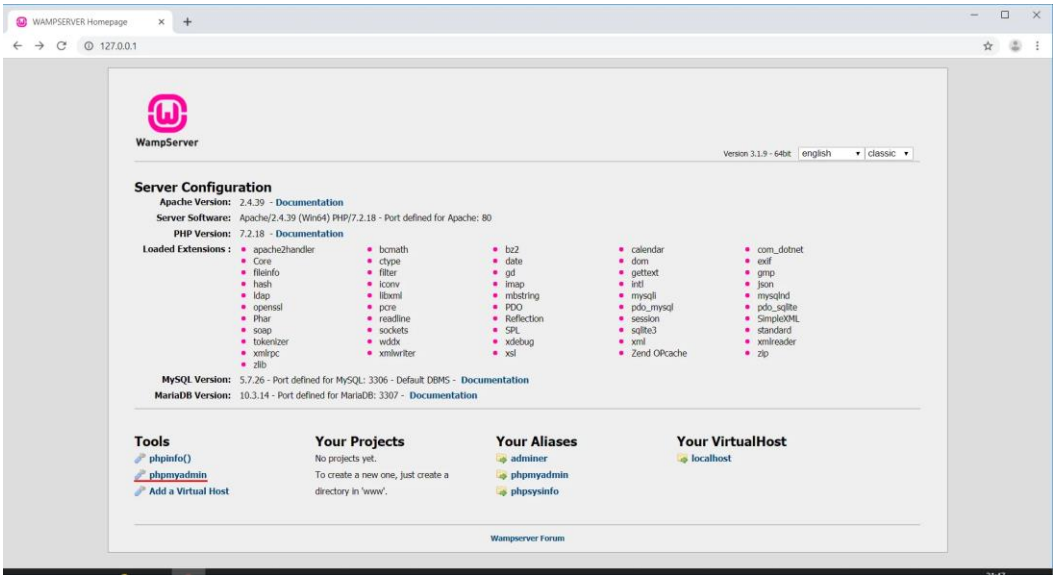


Рис. 15. Главная страница WAMP

После успешной авторизации можно добавить пользователей, если в этом есть необходимость, а также изменить пароль пользователя root. Перейдем на вкладку “Учетные записи пользователей”. В строке пользователя root кликнем по редактировать привилегии (рис. 16).

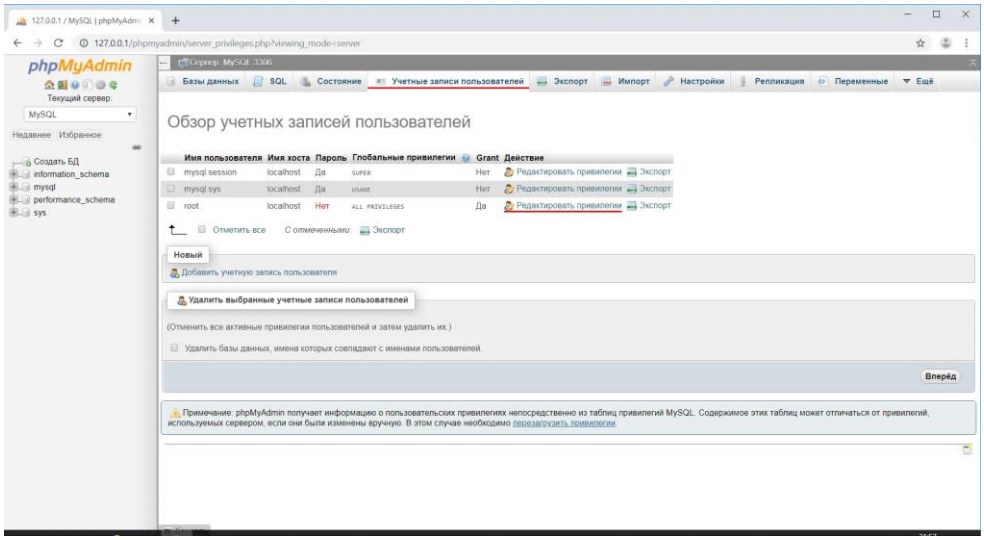


Рис. 16. Настройка привилегий пользователей

Страница обновится, кликаем по “Изменить пароль”. Вводим пароль и его подтверждение. Кликаем по кнопке “Вперед”.

Важно понимать. что root является пользователем с наивысшими привилегиями. Пароль должен быть сложным.

По окончании настроек перезапускаем сервер аналогично тому, как это было показано выше.

Последовательность выполнения лабораторной работы

1. Загрузить WAMP или подобный пакет.

2. Установить и настроить на хост со своим именем.

3. Разработать и проверить тестовую страницу.

4. Создать простую таблицу в MySQL.

5. Разработать и проверить Web-страницу, показывающую содержимое таблицы.

6. Показать результаты преподавателю.

7. Ответить на вопросы преподавателя.

Лабораторная работа №3 «Установка и настройка готовой CMS WordPress»

Теоретические сведения

WordPress — популярная CMS для наполнения сайтов. Предусмотрительные хостинг-провайдеры, такие как AdminVPS, предлагают инструменты для простой и быстрой установки WordPress на хостинг, что значительно упрощает жизнь пользователям. Однако вы можете установить WordPress и самостоятельно — и это гораздо проще, чем может показаться сначала.

Зачем нужна CMS.

Система управления контентом (CMS) — это программное обеспечение, которое упрощает создание и администрирование веб-сайта без необходимости владеть навыками программирования. WordPress, к примеру, является одной из самых популярных CMS, позволяющей даже новичкам быстро запустить собственный сайт или блог. С её помощью можно легко добавлять статьи, изображения и другие материалы, а также менять дизайн с помощью готовых шаблонов.

Преимущества CMS WordPress.

WordPress — это бесплатная система управления контентом с открытым исходным кодом, которая позволяет каждому создать собственный сайт или блог без навыков программирования. Её основные преимущества — простота использования, широкий выбор бесплатных тем и плагинов для кастомизации, а также активное сообщество, готовое помочь советом. Вместо того чтобы тратить значительные суммы на разработку сайта с нуля, вы можете самостоятельно запустить профессиональный веб-ресурс на WordPress. Благодаря поддержке хороших российских хостинг-провайдеров, таких как AdminVPS, установка и настройка становятся ещё проще и доступнее.

WordPress стал популярным благодаря своей простоте и гибкости. Эта CMS позволяет даже новичкам создать профессиональный сайт без знаний программирования. Огромный выбор бесплатных тем и плагинов даёт возможность настроить сайт под любые нужды. Активное сообщество и поддержка на русском языке делают работу с WordPress ещё более комфортной. Российские и международные компании, присутствующие в России, такие как «Лаборатория Касперского» и Microsoft, выбирают WordPress за его надёжность и экономичность, что помогает им существенно сэкономить средства в рублях на разработке сайтов.

Шаги перед установкой.

Перед тем как приступить к установке WordPress, нужно выполнить несколько подготовительных действий.

Для начала вам необходим:

- доступ к веб-ресурсу через FTP или shell,
- текстовый редактор,
- FTP-клиент (используется для установки «Вордпресса» на удалённый сервер).
- веб-обозреватель.

Выполните следующие шаги:

1. Убедитесь, что ваш хостинг или сервер соответствует минимальным требованиям «Вордпресса».
2. Скачайте последнюю версию WordPress с официального ресурса.
3. Извлеките файлы в папку.
4. Держите инструкции под рукой: распечатайте шаги установки, чтобы не потеряться.

Если вам нужно легко и быстро установить WordPress, этот метод идеально подойдёт. Он предназначен для пользователей с опытом веб-разработки или администрирования серверов. Знание того, как использовать FTP-клиент, настраивать файловую структуру и изменять конфигурационные файлы на сервере является обязательным, так как ручная установка требует уверенного владения этими навыками.

Во время ручной установки пользователи получают полный контроль над процессом, что позволяет гибко настраивать платформу в соответствии с требованиями проекта и избегать ограничений, которые могут возникать при автоматической установке через хостинг-панели.

Быстрая установка WordPress позволяет запустить ваш веб-проект в кратчайшие сроки и без лишних хлопот. Для этого вам нужен актуальный дистрибутив WordPress, который можно бесплатно скачать с официального сайта CMS. Этот метод особенно подходит тем, кто уже знаком с веб-технологиями и хочет сэкономить время.

1. Перейдите на сайт WordPress и загрузите актуальный установщик. Сохраните его в удобной папке на своём ПК и распакуйте архив. Это подготовит файлы для дальнейшей загрузки на сервер.
2. Следующим шагом будет создание репозитория, где будет храниться весь контент сайта. В панели управления хостинга, например,

ispmanager, найдите раздел «Базы данных» и создайте новую БД, выбрав имя и данные для входа. Убедитесь, что юзер имеет все необходимые права для доступа к репозиторию. Эти параметры понадобятся вам для настройки WordPress.

3. После распаковки WordPress в папке будут различные файлы, в том числе и файл wp-config-sample.php. Переименуйте его в wp-config.php — это конфигурационный файл, который вы заполните данными для связи WordPress с базой данных.

4. Откройте wp-config.php в текстовом редакторе, таком как Notepad++ или Sublime Text, и укажите данные для подключения к БД: наименование репозитория, имя пользователя, пароль и хост (обычно это localhost). Сохраните изменения. Правильное указание этих данных обеспечит успешное подключение WordPress к базе данных и корректную работу сайта.

5. Теперь пришло время перенести файлы WordPress на сервер. Для этого вам понадобится FTP-клиент (например, FileZilla). Решите, куда именно вы хотите установить WordPress:

- Если установка в корень сайта (например, *https://вашдомен.ру/*): скопируйте все файлы из папки WordPress в основную папку на сервере.

- Если установка в отдельную директорию (например, *https://вашдомен.ру/blog/*): переименуйте папку WordPress в желаемое имя (например, blog) и загрузите её в корневую директорию сервера. Это даст возможность разместить сайт WordPress как часть вашего веб-ресурса.

6. Когда файлы загружены, откройте браузер и перейдите на ваш сайт по одному из следующих адресов, в зависимости от выбранной папки установки:

- Для установки в корень сайта: *https://вашдомен.ру/wp-admin/install.php*

- Для установки в папку: *https://вашдомен.ру/имя_папки/wp-admin/install.php*. Не забудьте заменить вашдомен.ру на ваш фактический домен.

7. Следуйте указаниям установщика WordPress: введите данные сайта, укажите логин и пароль администратора. Завершив установку, вы получите подтверждение успешного завершения и сможете войти в административную панель.

Теперь WordPress установлен, и вы можете начать наполнять ваш сайт контентом, выбирать темы и устанавливать плагины для расширения функционала.

Последовательность выполнения лабораторной работы

1. Загрузить пакет установки WordPress.
2. Установить и настроить WordPress.
3. Разработать и проверить главную страницу сайта.
4. Показать результаты преподавателю.
5. Ответить на вопросы преподавателя.

Лабораторная работа №4 «Установка системы функционирования технической поддержки Hesk»

Теоретические сведения

Бесплатная версия Hesk доступна для веб-хостинга с поддержкой PHP и MySQL (PHP 5.3.0+, MySQL 5.0.7+).

Шаги установки Hesk.

1. Так как справочная служба работает при помощи веб-интерфейса, то нам понадобиться веб-хостинг с поддержкой php и mysql (PHP 5.3.0+, MySQL 5.0.7+). Скачиваем дистрибутив с официального сайта и языковой пакет.

Это будет два архивных файла: hesk311.zip и ru.zip.

2. Закачиваем содержимое архива **hesk311.zip** в корневую директорию хостинга, предназначенного для размещения файлов сайта.

Содержимое архива **ru.zip** в директорию language.

3. Создаем базу данных средствами хостинга и запоминаем параметры подключения.

4. Запускаем браузер и переходим по адресу нашей будущей службы технической поддержки в директорию **install**. Например **https://<ваш домен>/install** и нажимаем кнопку «**Click here to INSTALL HESK**» (рис. 17).

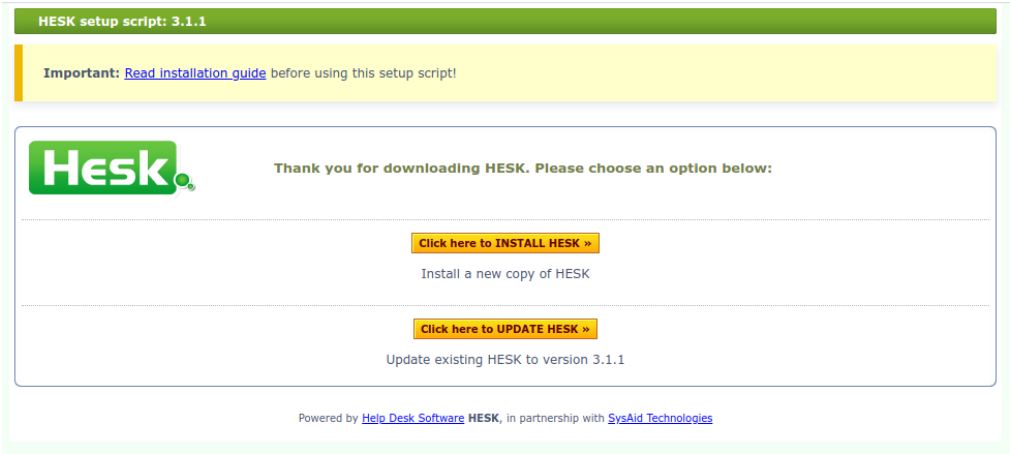


Рис. 17. Страница установки Hesk

5. Соглашаемся с Лицензией продукта.

6. Вводим параметры подключения к серверу баз данных, указываем логин и пароль администратора и часовой пояс.

7. Инсталлятор просит нас удалить папку install, запомнить логин и пароль администратора системы (рис. 18).

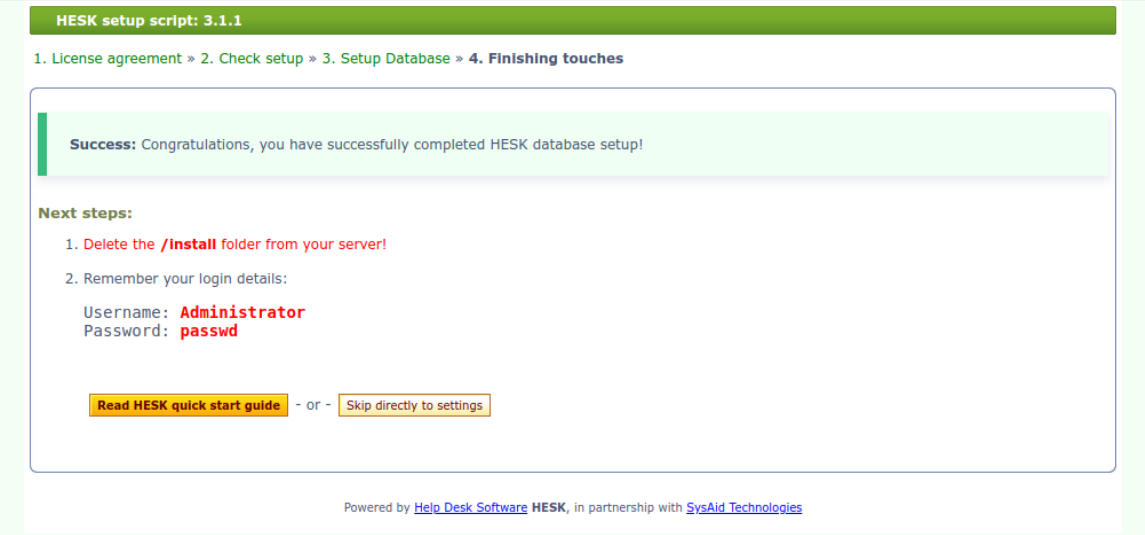


Рис. 18. Смена пароля администратора Hesk

8. Удаляем директорию Install на веб-хостинге и нажимаем кнопку «Skip dierctiy to settings». Переходим в панель управления справочной службы.

Настройка Hesk.

1. При первом запуске мы сразу попадаем в раздел «Settings». Здесь мы задаем все основные параметры нашей справочной службы. Для начала выберем русскоязычный интерфейс в подразделе «Default Language» (пакет с русским языком мы уже поместили в папку «language» на веб-хостинге) и нажимаем кнопку "Save changes".

2. После того как мы установили русский язык по умолчанию, переходим к основным настройкам панели. Указываем которые мы хотим поменять и сохраняем изменения.

3. Если мы хотим, чтобы сотрудникам службы технической поддержки и клиентам поступали оповещения по e-mail, то переходим в соответствующий раздел (E-mail) и указываем настройки подключения к smtp-серверу. Проверяем настройки подключения (рис. 19).

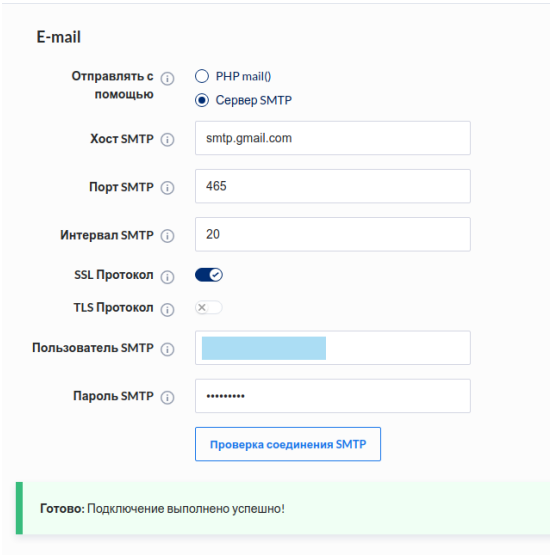


Рис. 19. Настройка E-mail в Hesk

4. После того как мы произвели ключевые настройки переходим к созданию категорий проблем, которые будет видеть пользователь на при создании заявки. Переходим в раздел «**Категории**», нажимаем кнопку «**Создать новую категорию**» и вносим название категорий, которые нам нужны (рис. 20).

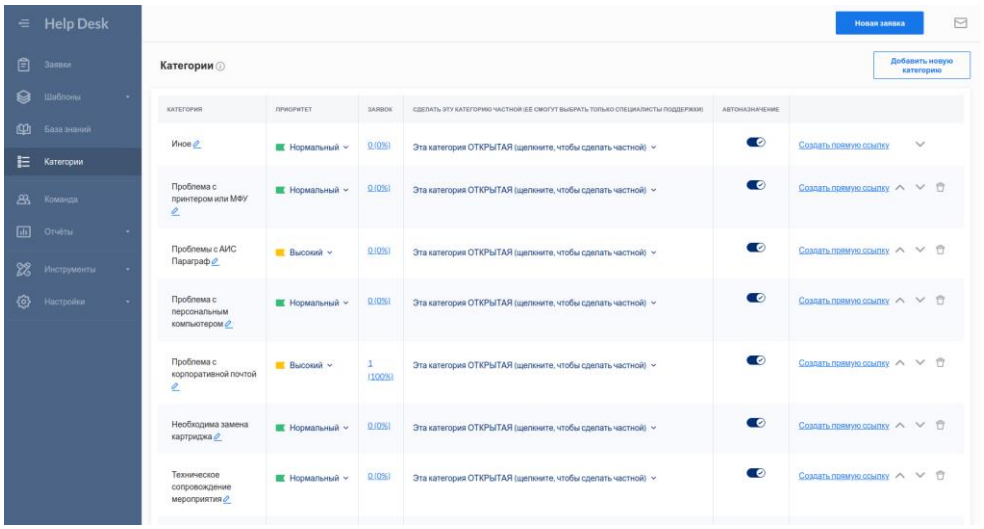


Рис. 20. Настройка категорий проблем

5. После того как мы создали **Категории**, переходим в раздел «**Команда**». Здесь мы создаем учетные записи сотрудников, которые будут помогать нам в работе службы технической поддержки пользователей. Это могут быть не только технические специалисты, но и методисты, которые могут проконсультировать по определённой теме (рис. 21).

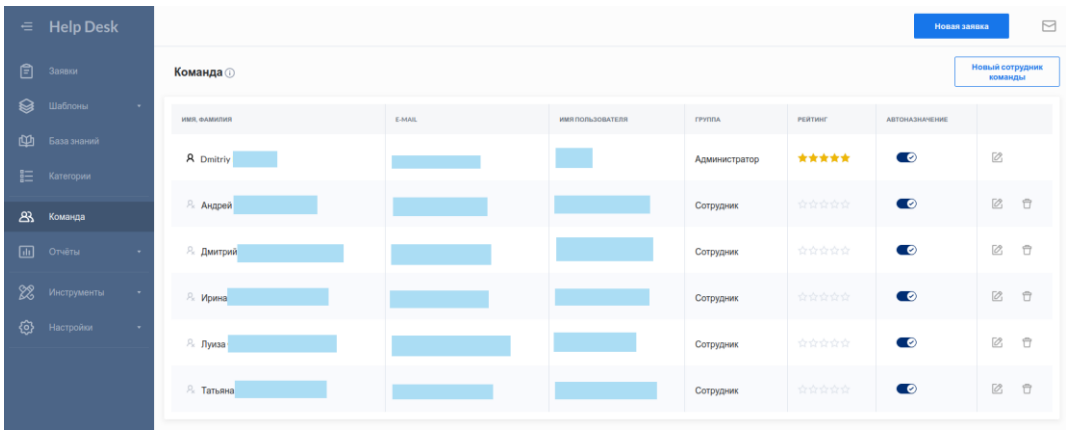


Рис. 21. Настройка команд

6. Не лишним будет наполнить базу знаний. Это позволит сэкономить время на консультации клиента (рис. 22).

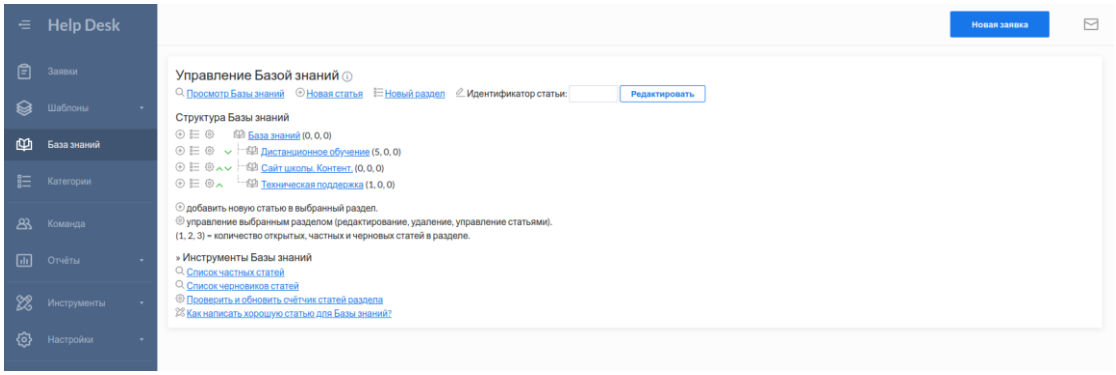


Рис. 22. Настройка базы знаний

На этом этап основной настройки админ-панели Hesk можно закончить. Теперь необходимо научить пользователей подавать заявки. Логично будет создать инструкцию и разместить ее в **Базе знаний**. Тогда можно будет легко сослаться на неё.

Инструкция для пользователя.

Со стороны пользователя работа с системой Hesk состоит из следующих шагов:

1. Уважаемый пользователь! Сервис технической поддержки доступен по адресу https://support.itsch.ru

2. Проверьте, может Ваша проблема уже описана в Базе знаний? Если Вы не нашли решения, то выберете раздел «Создать новую заявку».

3. Выберете категорию, которая наиболее полно характеризует возникшую у Вас проблему (рис. 23).

4. Заполняем поля формы заявки:

- Имя, фамилия.

- E-mail.
- Приоритет — необходимо выбрать.
- Тема — краткое описание проблемы.
- Сообщение — описание возникшей проблемы. Максимально подробно.
- Прикрепить файл — возможность прикрепить файла (например фото проблемы).
- Указать, что Вы не робот.
- Нажать кнопку «Создать заявку».

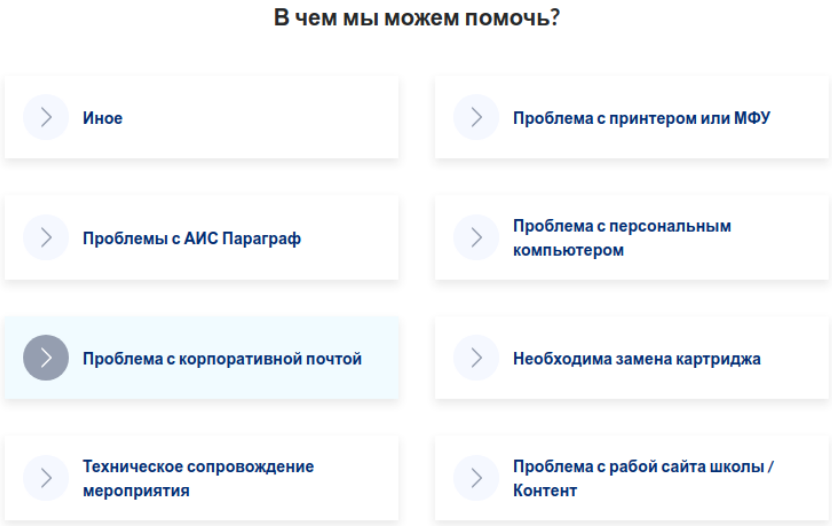


Рис. 23. Выбор пользователем категории заявки

5. Оповещение о создании заявки будет выслано на указанную указанный ящик электронной почты.

6. Состояние созданной заявки можно проверить при помощи функции «Найти уже созданную».

7. Ввести идентификатор заявки и адрес электронного почтового ящика, который был указан в заявке (рис. 24).

8. После нажатия кнопки «Посмотреть заявку» появится возможность просмотра статуса заявки и диалога с сотрудником технической поддержки (рис. 25).

9. Ждем Ваших заявок.

После всех проделанных процедур размещаем ссылку на Help Desk в Цифровой корпоративной среде (чат, корпоративный сайт) и готовимся продуктивной работе.

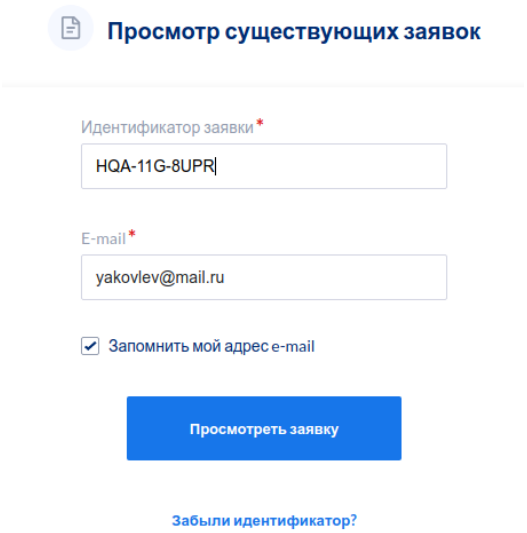


Рис. 24. Ввод идентификатора заявки

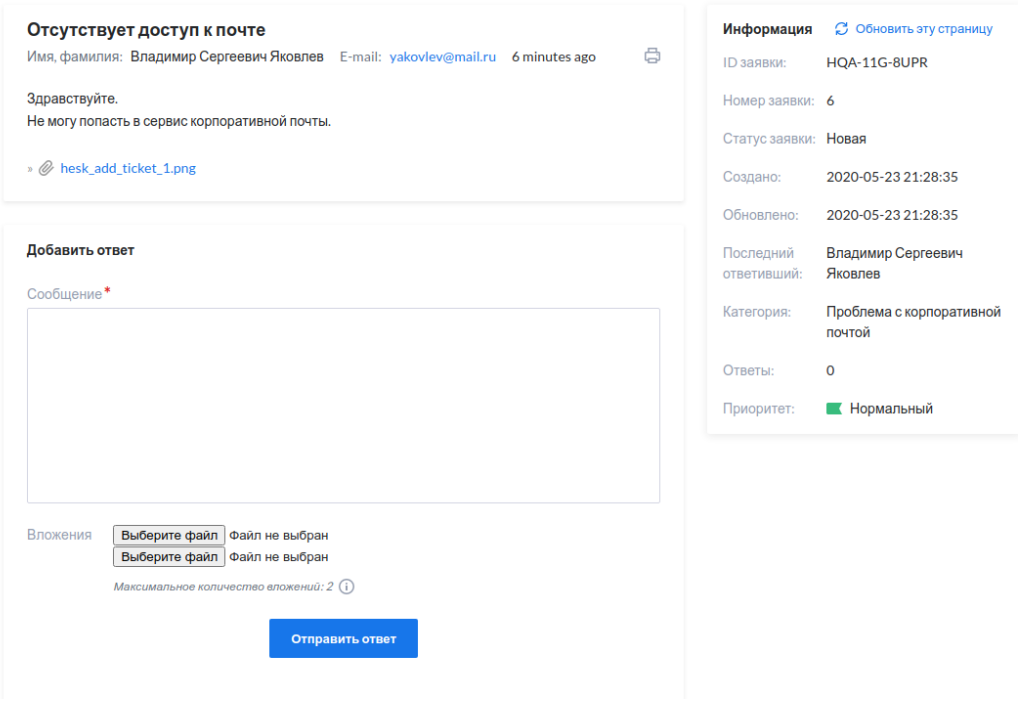


Рис. 25. Просмотр результата заявки

Последовательность выполнения лабораторной работы

1. Загрузить пакет установки Hesk.
2. Установить и настроить Hesk.
3. Добавить несколько категорий заявок и базу знаний.
4. Имитировать подачу заявки пользователя.
5. Имитировать ответ сотрудника службы поддержки.

6. Показать результаты работы преподавателю.
7. Ответить на вопросы преподавателя.

Лабораторная работа №5 «Установка среды веб-разработки на базе VS Code»

Теоретические сведения

Visual Studio Code (VS Code) является одной из наиболее популярных сред веб-разработки. Последовательность установки состоит из следующих шагов:

Шаг 1: Скачивание установочного файла.

Откройте официальный сайт Visual Studio Code: `https://code.visualstudio.com/;`

1. Нажмите кнопку "Download for Windows" (или соответствующую для вашей операционной системы);
2. Система автоматически предложит версию, подходящую для вашей ОС. Если нужна другая версия, нажмите на стрелку рядом с кнопкой загрузки и выберите нужный вариант;
3. Сохраните установочный файл в удобное место на вашем компьютере.

Шаг 2: Запуск установки.

1. Найдите скачанный файл (обычно в папке "Загрузки") и запустите его двойным щелчком;
2. Если появится предупреждение безопасности, подтвердите, что хотите запустить программу;
3. Примите лицензионное соглашение, нажав "I accept the agreement" и кнопку "Next";
4. Выберите папку для установки (можно оставить предложенную по умолчанию) и нажмите "Next".

Шаг 3: Настройка параметров установки.

1. На экране "Select Additional Tasks" рекомендую отметить все опции:

- "Create a desktop icon" — создать ярлык на рабочем столе
- "Add to PATH" — добавить в переменную PATH (позволит запускать VS Code из командной строки)
- "Register Code as an editor for supported file types" — зарегистрировать как редактор для поддерживаемых типов файлов

2. Нажмите "Next", а затем "Install" для начала установки.

Шаг 4: Завершение установки.

1. Дождитесь окончания процесса установки;
2. На финальном экране оставьте отмеченным пункт "Launch Visual Studio Code" и нажмите "Finish";
3. VS Code запустится автоматически.

Шаг 5: Первый запуск и проверка.

1. При первом запуске VS Code может предложить настройку языка интерфейса — выберите предпочтительный;
2. Создайте новый файл через меню File → New File (или Ctrl+N);
3. Напишите простой код, например: `console.log('Hello, World!');`
4. Сохраните файл с расширением .js: File → Save (или Ctrl+S);
5. Если подсветка синтаксиса работает, значит IDE установлена корректно.

Базовая настройка IDE для комфортной работы.

Давайте разберем ключевые настройки, которые сделают вашу работу продуктивнее.

1. Настройка темы и шрифта:

Хороший шрифт и контрастная тема могут существенно снизить нагрузку на глаза при длительном кодировании:

- В VS Code: File → Preferences → Color Theme (или Ctrl+K Ctrl+T)
- Популярные темы для новичков: "Dark+", "Light+" (встроенные), "One Dark Pro", "Material Theme" (устанавливаются отдельно)
- Для настройки шрифта: File → Preferences → Settings → Text Editor → Font
- Рекомендуемые шрифты для программирования: Fira Code, JetBrains Mono, Source Code Pro (поддерживают лигатуры для более читаемого кода)

2. Настройка автосохранения и форматирования:

Эти настройки помогут автоматизировать рутинные действия:

- Включение автосохранения: File → Preferences → Settings → Files: Auto Save → выберите "afterDelay" или "onFocusChange"

- Настройка автоформатирования: File → Preferences → Settings → Text Editor → Formatting → Format On Save (установите флажок)
- Для Python в VS Code: установите расширение "Python" от Microsoft и включите линтер (PEP8) через Settings
- Для JavaScript: установите ESLint и Prettier для автоматического форматирования кода.

3. Настройка автодополнения и подсказок:

- В VS Code автодополнение работает "из коробки", но можно улучшить его работу
- Настройка задержки появления подсказок: File → Preferences → Settings → Editor → Suggest → Delay
- Для более умного автодополнения установите расширения для вашего языка (например, "Python IntelliSense" для Python)
- Включите параметр "Accept Suggestion On Commit Character" для автоматического принятия подсказок при вводе определенных символов (точка, скобка и т.д.)

4. Настройка отладчика:

Отладчик — ваш лучший друг в поиске ошибок:

- Установите расширение для отладки вашего языка (для Python это встроено в расширение Python)
- Настройте точки останова: кликните на поле слева от номера строки или нажмите F9
- Запустите отладку через меню Run → Start Debugging (F5) или создайте конфигурацию отладки в файле launch.json
- Изучите панель отладки: переменные, стек вызовов, точки останова, наблюдение за выражениями

5. Настройка терминала:

Встроенный терминал избавляет от необходимости переключаться между окнами:

- Открыть терминал: View → Terminal или Ctrl+` (обратная кавычка)
- Изменить тип терминала: нажмите на стрелку вниз рядом с "+" в панели терминала и выберите предпочтительный shell (PowerShell, Command Prompt, Git Bash)

- Настройка внешнего вида: File → Preferences → Settings → Terminal

Вот базовый набор настроек, которые сделают работу новичка комфортнее. Вы можете применить их, добавив в файл settings.json (File → Preferences → Settings → значок {} в правом верхнем углу):

```
{
"editor.fontSize": 14,
"editor.fontFamily": "Fira Code, Consolas, monospace",
"editor.fontLigatures": true,
"editor.wordWrap": "on",
"editor.minimap.enabled": false,
"editor.formatOnSave": true,
"editor.tabSize": 4,
"files.autoSave": "afterDelay",
"files.autoSaveDelay": 1000,
"workbench.colorTheme": "One Dark Pro",
"terminal.integrated.fontSize": 14,
"editor.suggestSelection": "first",
"workbench.startupEditor": "newUntitledFile"
}
```

Интеграция в IDE: плагины и дополнения.

Базовая функциональность IDE — это только начало. Настоящая сила интегрированных сред разработки раскрывается с установкой правильных плагинов. Представьте, что ваша IDE — это смартфон, а плагины — приложения, которые превращают его из простого устройства для звонков в многофункциональный инструмент.

Интеграция правильных плагинов в вашу IDE может:

- Автоматизировать рутинные операции;
- Добавить поддержку новых языков и фреймворков;
- Улучшить качество кода через статический анализ;

- Добавить интеграцию с внешними сервисами;
- Кастомизировать интерфейс под ваши потребности.

Чтобы плагин в Visual Studio Code:

1. Нажмите на иконку расширений в боковой панели (или Ctrl+Shift+X);
2. В поиске введите название нужного расширения;
3. Нажмите "Install" рядом с выбранным расширением;
4. Некоторые расширения требуют перезапуска IDE — следуйте инструкциям на экране;
5. После установки настройте расширение через его параметры (если необходимо).

Основные плагины для веб-разработки (табл. 1):

Таблица 1. Плагины VS Code для веб-разработки

Название плагина	**Назначение**
Live Server	Запуск локального сервера с автообновлением страницы при изменениях
JavaScript (ES6) code snippets	Типовые фрагменты кода Javscript
npm Intellisense	Интеллектуальный подсказчик кода
SASS plugin	Автоматическая компилляция SASS.

Интеграция с системами контроля версий.

Работа с Git — неотъемлемая часть современной разработки, даже для новичков:

1. Установите расширение "Git History" или "GitLens" для расширенной интеграции с Git;
2. Настройте авторизацию в GitHub/GitLab через расширение "GitHub Pull Requests";
3. Используйте визуальный интерфейс для коммитов, веток и слияний через встроенный Source Control (Ctrl+Shift+G);

4. Настройте игнорируемые файлы через .gitignore (расширение "gitignore" может помочь).

Автоматизация рабочего процесса:

По мере роста ваших навыков, стоит настроить автоматизацию рутинных задач:

- Установите Task Explorer для запуска задач через интерфейс
- Создайте собственные сниппеты кода: File → Preferences → User Snippets
- Настройте запуск тестов через Test Explorer UI
- Используйте расширение Settings Sync для синхронизации настроек между разными компьютерами

Важно помнить, что не нужно устанавливать все плагины сразу. Начните с базового набора и добавляйте новые по мере необходимости. Слишком много расширений могут замедлить работу IDE и создать путаницу в интерфейсе.

Регулярно проверяйте обновления установленных плагинов через вкладку "Extensions" — разработчики часто исправляют ошибки и добавляют новые возможности. Также полезно периодически просматривать список самых популярных расширений в официальном маркетплейсе — там можно найти новые полезные инструменты.

Последовательность выполнения лабораторной работы

1. Загрузить и установить VS Code.
2. Выполнить настройку среды по своим предпочтениям.
3. Добавить основные плагины: LiveServer и др.
4. Разработать проверочную страницу.
5. Показать результаты работы преподавателю.
6. Ответить на вопросы преподавателя.

Лабораторная работа №6 «Публикация веб-приложения на хостингах разного типа»

Теоретические сведения

Рассмотрим публикацию веб-приложения на одном из наиболее популярных бесплатных хостингов – GitHub Pages.

GitHub — это популярная веб-платформа для публикации IT-проектов в интернете, совместной работы и т. д. В 2022 году на неё выложили более 3,5 млрд материалов. Для размещения кода в интернете разработчики добавили GitHub Pages. С их помощью программисты создают сайты для расположения рабочих проектов, портфолио и др. При этом любой пользователь может получить к ним доступ и использовать готовые наработки для обучения или других задач.

GitHub Pages — одна из лучших возможностей GitHub. Сервис позволяет размещать статические веб-страницы прямо из репозитория GitHub. Это означает, что вы можете использовать свой репозиторий для хранения кода и файлов веб-ресурса. GitHub автоматически опубликует их как интернет-сайт, к которому можно получить онлайн-доступ.

Существует три типа сайтов GitHub Pages:

- Проект. Он подключён к определённому публичному репозиторию, который размещён на GitHub (к примеру, библиотека JavaScript или коллекция рецептов).
- Пользователь.
- Организация.

Последние два вида сайтов подключают к учётным записям на GitHub.com.

Чтобы опубликовать сайт пользователя, необходимо создать репозиторий, привязанный к личной учётной записи с именем <username>.github.io. Для размещения веб-ресурса компании — репозиторий, принадлежащий организации с именем <organization>.github.io. Если вы не используете персональный домен, страницы пользователей и организаций доступны по адресу http(s)://<username>.github.io или http(s)://<organization>.github.io.

Исходные файлы для сайта проекта размещают в том же репозитории, что и их проект. При отсутствии персонального домена сайты проекта доступны по адресам:

- http(s)://<username>.github.io/<repository>;

- http(s)://<organization>.github.io/<repository>.

Стоит отметить, что одновременно пользователь может загрузить на хостинг-сервер только один сайт, привязанный к учётной записи. Количество веб-ресурсов для проектов не ограничено.

Плюсы и минусы GitHub Pages.

Размещение на GitHubPages — распространённая практика в IT-компаниях и среди фрилансеров. Благодаря хостинг-сервису можно упростить совместную работу, создать портфолио из рабочих файлов и так далее.

Преимущества GitHub Pages:

- Бесплатно. Частные пользователи могут создавать сайты без покупки расширенной подписки.
- Быстрая и простая реализация. Благодаря поддержке статического HTML, а не CMS или сложных серверных решений создание нового сайта и структуры не займёт много времени. Обмен данными будет осуществляться только между хостингом и пользовательским устройством. Базы данных и тому подобное при статистическом HTML не нужны.

Кроме того, GitHub управляет тонкостями хостинга, что ещё больше повышает скорость. Наконец, внесение изменений и развёртывание из репозиториев происходит быстро и без проблем.

Главный недостаток сервиса GitHub Pages вытекает из его же преимущества — статического веб-сайта. Отсутствие динамики накладывает на проект ряд ограничений:

- Функциональность, подобная серверной части, должна быть реализована сторонними решениями.
- Приходится создавать новый сайт при изменении контента.
- Нет динамической функциональности.

Кроме того, в GitHub нельзя создавать сайты для коммерческих проектов, например площадок e-commerce.

Другие ограничения:

- Рекомендуемый объём репозиториев исходных текстов составляет 1 ГБ.
- Максимальный размер опубликованных сайтов не превышает 1 ГБ.

- Сайты имеют лимит по пропускной способности в 100 ГБ в месяц.
- Для сайтов установлено ограничение в 10 сборок в час.

Разберём, как создавать и публиковать сайты на GHB. Пошаговая инструкция:

1. Создайте репозиторий GitHub.

Войдите в свою учётную запись GitHub и создайте новый общедоступный репозиторий с любым названием, желательно отражающим суть проекта (рис. 26).

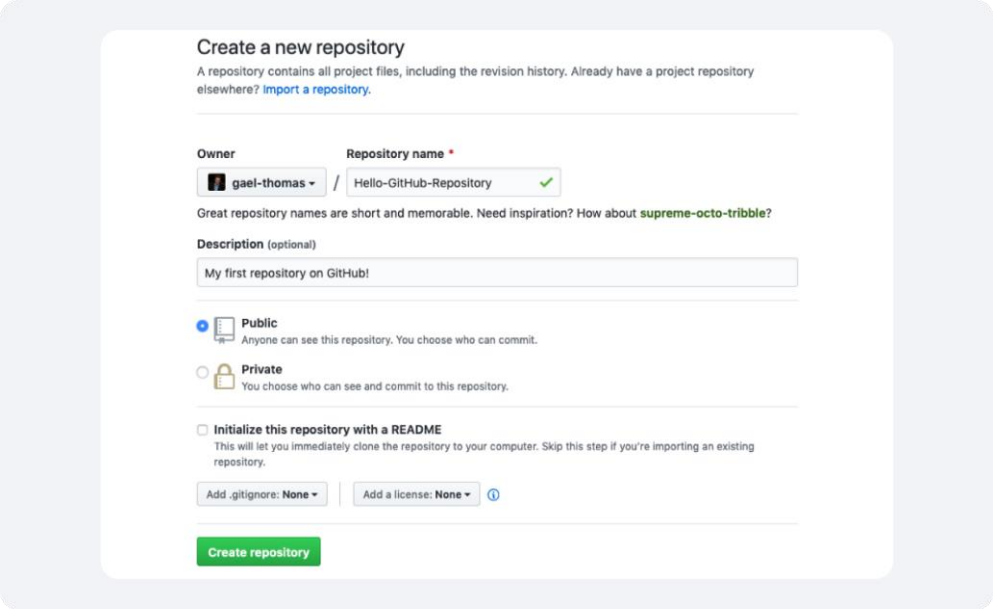


Рис. 26. Создание репозитория GitHub

Если компания хочет ограничить доступ к проекту, то стоит кликнуть на кнопку Private. Тогда пользователи смогут открыть репозиторий только после разрешения.

2. Превратите репозиторий в сайт.

Чтобы опубликовать программный код в сети, преобразовываем репозиторий в сайт с уникальным адресом. Открываем проект и переходим в настройки (рис. 27).

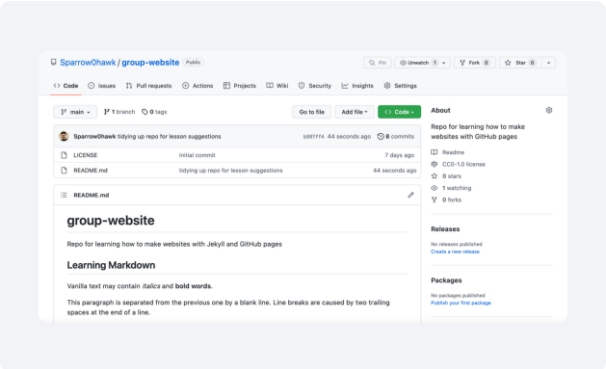


Рис. 27. Преобразование репозитрия GitHub в сайт

В меню слева нажимаем на кнопку Pages и переходим к Branch. Если это сайт пользователя или компании, то выбираем ветку Main. Если организация работает над множеством проектов в одном репозитории, нужно создать несколько веток для них. Это покажет сайту, откуда именно брать информацию. Нажимаем на Save. В течение нескольких минут на странице появится ссылка, ведущая на веб-ресурс. Если ничего не произошло, обновите страницу. Готово — ваш сайт доступен по адресу "your_nickname.github.io/your_repository_name" (рис. 28).

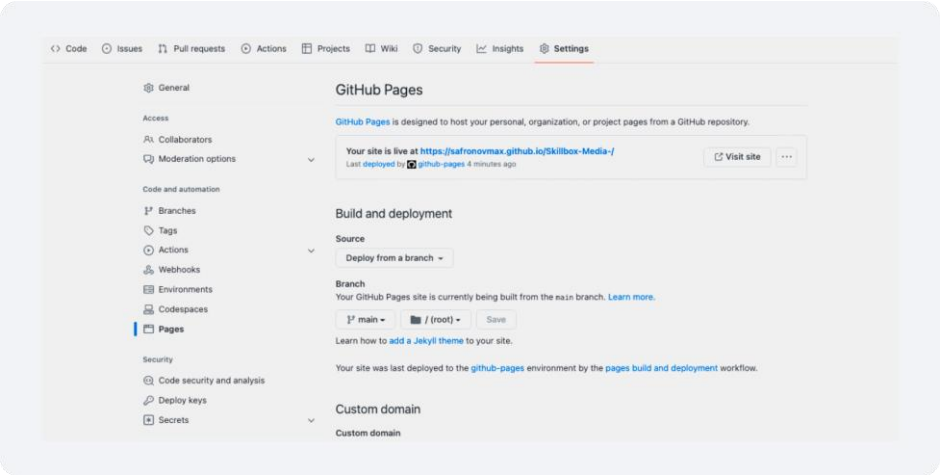


Рис. 28. Настройка сайта

Теперь пользователь может перейти на сайт. Правда, пока он будет состоять из одного файла (нет даже интерфейса), поэтому оставлять ссылку в портфолио точно не стоит.

3. Сделайте сайт красивым .

Можно использовать бесплатные темы от Jekyll. Текст можно оформить с помощью Markdown — облегчённого языка разметки. Он создан для форматирования текстовых документов, с максимальным сохранением читаемости, и подходит для преобразования в языки HTML, Rich Text и другие.

Инструкция, как сделать интерфейс красивым:

1. Заходим на страницу поддерживаемых тем GitHub и выбираем подходящий вариант. В нашем случае использован стиль Cayman. С ним сайт выглядит современно и при этом не перегружен деталями.

2. В документе Readme копируем строку remote_theme: название_темы (рис. 29).

3. Открываем репозиторий ресурса и создаём новый документ _config.yml. Для этого нужно кликнуть на Add file, а затем на кнопку Create new file.

Рис. 29. Подключение темы

4. Вставляем взятую из ReadMe строку и сохраняем документ.

Через несколько секунд сервис изменит оформление. Сайт уже выглядит неплохо, но всё ещё пустой. Остаётся добавить текст и другую информацию. Важно: при использовании Jekyll для наполнения страницы необходимо освоить разметку Markdown. Она простая, так что проблем с изучением не будет. Разметку подробно описали разработчики Microsoft. Разберём пару примеров:

- Выделение жирным — *текст*.
- Заголовок первого уровня — #текст#, второго — ##текст## и т. д.

Публикация веб-сайта на типовом хостинг-провайдере.

Развернуть сайт на хостинге — значит загрузить файлы и базу данных в определенный сервис, который предоставляет услуги по размещению веб-проектов. Как итог, мы получим работающий ресурс.

Сайты обычно состоят из двух частей: файлы и база данных. Если речь идет о ленде, то часто они поставляются без системы управления контентом и работают на голом HTML+JS. В этом случае возиться с импортом базы данных не нужно.

Новичкам в сайтостроении важно понимать, что все хостинг-провайдеры по факту предоставляют одинаковые услуги. Они предлагают клиентам место на своем оборудовании и зарабатывают на этом деньги.

Представьте, что вы купили дом с 10-ю квартирами и решили их сдать. Каждое помещение является самостоятельным, но при этом входит в состав основного здания. У квартир общая система отопления, проводка и водопровод. Хостинг работает аналогичным образом.

Чтобы залить сайт на хостинг, необходимо подготовить файлы, купить домен и определиться с потенциальной нагрузкой на сервер. Каждая задача из списка важна, потому что от нее зависит конечный результат.

Некоторые арбитражники сначала регистрируются у 10 разных хостинг-провайдеров, а только потом покупают домен, скачивают лендинг в партнерке и выполняют другие подготовительные мероприятия. Эту стратегию нельзя назвать провальной, но лучше делать все по порядку.

Первый шаг — покупка домена. Если сайт создается под пролив конкретного оффера, лучше выбрать тематический адрес. Зарегистрировать его можно у любого популярного регистратора: Porkbun, AHnames, GoDaddy и т. д.

Второй шаг — подключение домена к сервису CloudFlare. Он позволяет скрыть реальный IP проекта, отсечь ботов и сэкономить ресурсы хостинга. Если решитесь использовать сервис, внимательно следите за работоспособностью сайта, потому что Роскомнадзор периодически блокирует айпишники.

Использование CloudFlare для многих арбитражников является обыденностью, но не все не понимают, какая от него польза. Кроме указанных выше плюсов, еще сервис может увеличить скорость загрузки ресурса благодаря кэшированию данных.

Также он позволяет установить бесплатный SSL-сертификат, чтобы контент загружался по защищенному протоколу (рис. 30). Если хотите обеспечить максимальное удобство для пользователей, не жалейте времени на настройку CloudFlare.

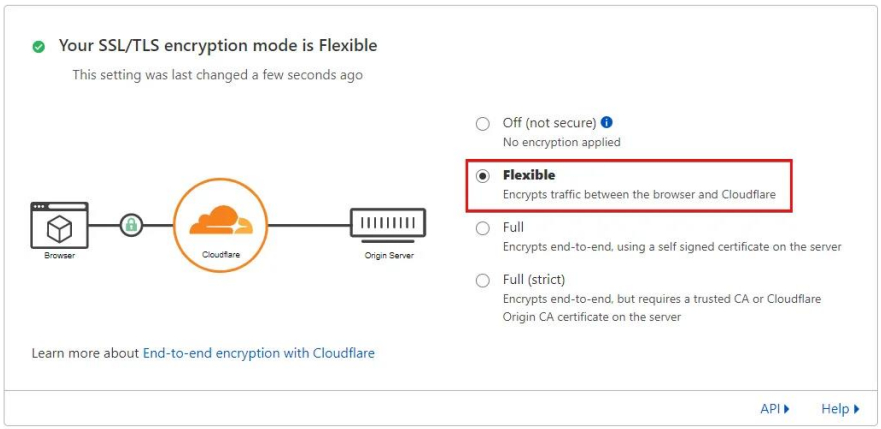


Рис. 30. Установка сертификата SSL

После того, как домен прикреплен к NS-серверам сервиса, наступает время подготовить файлы и базу данных. Загрузите шаблон в партнерке или создайте свой на основе любого темплейта из интернета.

Как только файлы будут готовы к заливке, используйте Winrar или любой другой софт для создания архива. Если сайт статический и не предполагает сохранение таблиц в базе данных, установка будет осуществляться в один этап.

Подход к выбору доменов для лендов у каждого арбитражника свой, но нельзя зацикливаться только на этой задаче. От качества хостинга и оптимизации загрузки контента зависит пользовательский опыт. Даже если сайт без проблем пройдет модерку в рекламной сетке, низкая скорость может перекрыть поток лидов.

В случае с многостраничным сайтом возни будет больше, но если не спешить, проект без проблем будет работать на новом хостинге. Главное, соблюдать базовые правила работы с MySQL: создание архива с расширением .sql.zip, одна кодировка, сжатие.

Если нужно перенести существующий WordPress сайт на другой хостинг, есть смысл использовать специализированные инструменты вроде Duplicator. Плагин создает «локальный снимок» проекта и дает возможность импортировать его на другом сервере.

Создание блога на WordPress или любая другая задача, связанная с сайтостроением, может показаться вебам чересчур сложной, но результат зависит исключительно от подхода. Разбираться в программировании совсем необязательно, базовые операции вроде заливки сайта чаще всего выполняются с первой попытки.

После успешной регистрации домена, подключения его к CloudFlare и выбора хостинга, остается выполнить главную задачу — установить сайт и убедиться в корректной работе.

У каждого арбитражника свой подход к заливке сайта. Некоторые используют ISPmanager, другие предпочитают юзать Duplicator, который берет на себя всю рутину и вебу остается только скачать архив на компьютер.

Ниже собрали все популярные способы заливки ленда или сайта. Можно использовать любой из них, но всегда стоит отталкиваться от особенностей сайта и хостинга. Например, некоторые провайдеры не дают доступ к SSH или ограничивают пропускную способность FTP.

ISPmanager, cPanel и другие интерфейсы для управления веб-сервером позволяют быстро и безболезненно выполнять любые технические операции. Чаще всего вебы загружают файлы сайтов через них.

Все, что нужно сделать — это загрузить запакованный архив с картинками и другими файлами в корень, а потом распаковать его. Чтобы сайт правильно работал, следите за отсутствием дополнительных папок в

корне. К примеру, если в главной директории будут не файлы сайта, а еще одна промежуточная папка, контент не загрузится.

Если сайт создан с использованием базы данных, придется импортировать бэкап MYSQL. Старую базу нужно запаковать в архив, затем создать новую базу на хостинге и выполнить операцию импорта (рис. 31).

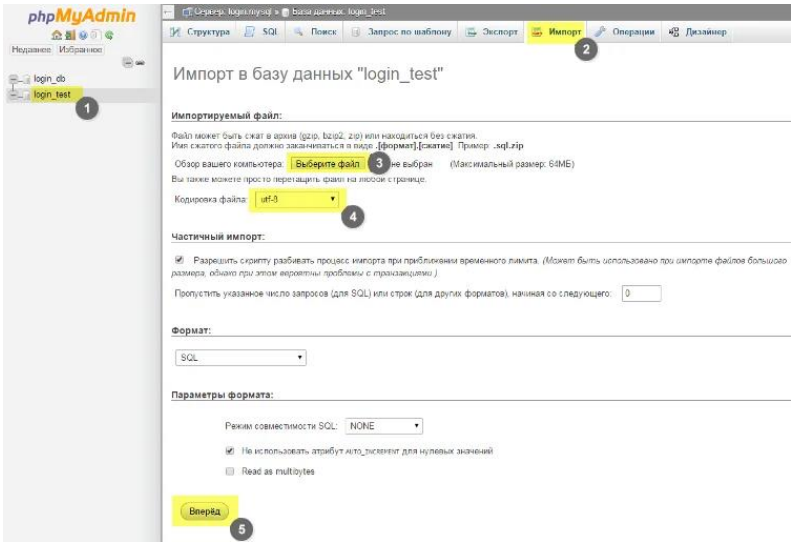


Рис. 31. Создание архива БД MySQL

На некоторых хостингах и конструкторах сайтов импорт БД реализован через стандартный интерфейс панели управлении. Это удобно, но загрузка данных не всегда проходит гладко.

Еще один популярный вариант заливки сайта подразумевает передачу данных по протоколу FTP. Для этого нужно создать учетную запись на хостинге и скачать любой подходящий клиент на компьютер.

Чаще всего вебы используют в качестве средства для передачи данных Filezilla. Учитывайте, что для переноса архивов на несколько гигабайт понадобится стабильное соединение. Если интернет работает через раз, с заливкой возникнут проблемы.

Через FTP в стандартном конфиге можно передать на хостинг только файлы, но при желании также есть возможность использовать связку FTP-MYSQL. Она подойдет тем, кто любит копаться в технических настройках.

Лучше всего передавать файлы по FTP в архиве, а потом распаковать его с помощью cPanel или другой панели управления хостингом. В процессе обработки архива можно заниматься другими задачами.

SSH — популярный сетевой протокол для выполнения операций на удаленном сервере. Большинство хостинг-провайдеров предоставляют возможность подключаться к оборудованию через него, но им пользуются преимущественно опытные вебы.

Главное преимущество SSH заключается в защищенности — между пользователем и сервером создается безопасный туннель. Еще с помощью консоли можно запускать различные веб-приложения.

У некоторых провайдеров доступ к серверу по этому протоколу позиционируется как дополнительная услуга. Если она включена в цену как подарок, можно расслабиться. А покупать ее отдельно не всегда выгодно из-за высокой цены.

Залить архив на хостинг несложно, но если речь идет не о ленде, а о сайте на базе популярной системы управления контентом, скорее всего, придется выполнять дополнительные задачи.

К примеру, часто приходится обновлять ссылки из-за того, что страницы загружаются по HTTP, несмотря на подключенный SSL-сертификат. Еще могут возникнуть проблемы с подключением базы данных.

Что касается других мероприятий, вроде вывода сайта в зеленую зону PageSpeed, их лучше доверять опытным разработчикам. Сжать скрипты можно и самостоятельно, но не будет гарантий стабильной работы. Можно использовать готовые SEO-шаблоны для WordPress, но лендингов среди них мало.

Не стесняйтесь обращаться в поддержку хостинга при возникновении любых проблем. Иногда неполадки появляются из-за особенностей конфигурации оборудования, а не отсутствия знаний у вебмастера.

Последовательность выполнения лабораторной работы

1. Разработать простой тестовый сайт из двух-трех страниц.

2. Создать репозиторий на GitHub.

3. Опубликовать сайт на GitHub Pages.

4. Опубликовать сайт на бесплатном хостинге.

5. Показать результаты работы преподавателю.

6. Ответить на вопросы преподавателя.

Лабораторная работа №7 «Составление блок-схемы работы оператора технической поддержки»

Теоретические сведения

Расскажем на примере нашей компании, как организовать техническую поддержку клиентов. Для удобства как клиента так и самой компании, мы организовали ИТ-поддержку по принципу «единого окна» – все обращения стекаются в одну точку и далее обрабатываются. Такой точкой является Ticket-система. Инциденты в системе обрабатываются следующими способами (рис. 32):

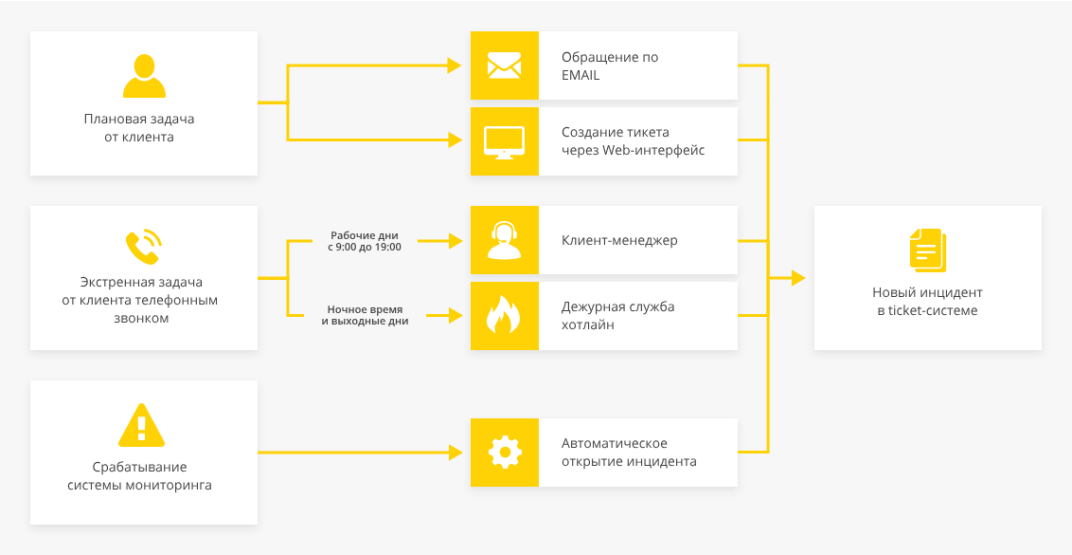


Рис. 32. Схема технической поддержки пользователей

Постановка задачи.

Web-интерфейс: Нужно заполнить форму на сайте. Здесь же возможно увидеть все свои задачи, статусы по ним, комментарии исполнителя и прочую информацию.

Email: Просто нужно отправить письмо на адрес системы с почты, которая указана при регистрации контрагента. По адресу отправителя будет происходить идентификация компании и конкретного ее сотрудника и далее инцидент будет передан закрепленному за клиентом ИТ-инженеру.

По телефону: Данный способ связи актуален для срочных обращений, когда нет возможности подать заявку другим способом. В этом случаи инцидент будет зарегистрирован сотрудником компании. В рабочее время обращения принимают клиент-менеджеры, в остальное время работает круглосуточная служба поддержки.

Проактивное реагирование на инцидент.

С целью обнаружить проблему ранее чем о ней сообщит клиент, в компании организована система проактивного реагирования на базе системы мониторинга Zabbix. В случаи срабатывания критического триггера в системе автоматически открывается инцидент. Данный механизм в ряде случаев позволяет не только среагировать, но и разрешить инцидент до обнаружения проблемы клиентом. Основная сложность заключается в правильной настройке системы мониторинга – нужно сделать так, чтобы система с одной стороны отслеживала наступление всех критических событий, а с другой стороны не создавала ложных инцидентов. В первое время после запуска системы у нас было просто огромное количество ложных срабатываний, так как система была настроена очень чувствительно.

За каждым клиентом закрепляется личный ИТ-инженер, который обрабатывает обращения клиента в рабочее время компании. В случае возникновения обращения на выходных с инцидентом работает дежурный ИТ-инженер, а в ночное время – служба круглосуточной поддержки. Если у дежурного инженера или круглосуточного оператора возникают трудности с разрешением инцидента – к его решению привлекается закрепленный инженер. Далее, если инженер не может разрешить инцидент, он поднимается на уровень руководителя модуля поддержки (TeamLead), то есть происходит горизонтальная эскалация. Если же у ТимЛида не достаточно организационных полномочий или ресурсов для решения задачи – происходит вертикальная эскалация инцидента и его разрешениям занимается руководитель ИТ-отдела. Схема показана на (рис. 33).

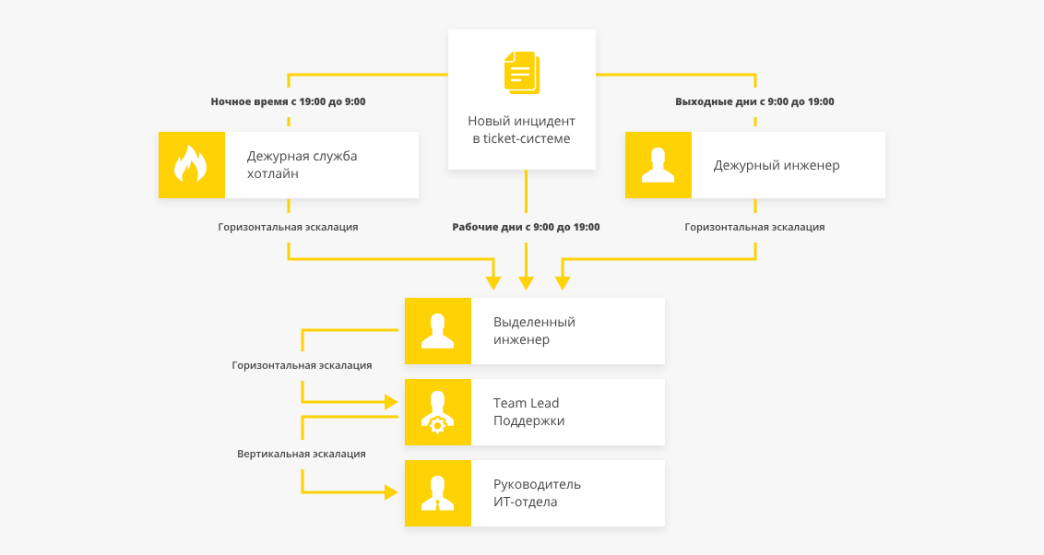


Рис. 33. Схема разрешения инцидента

Удовлетворенность клиента решением его задачи является важнейшей задачей работы ИТ-отдела, поэтому инцидент закрывается только после соответствующего подтверждения клиента. Если заказчик не доволен решением – инцидент возвращается на доработку. В случае если клиент

вернул на доработку задачу более 2-х раз – она возвращается не инженеру, а руководителю ИТ-отдела. Схема показана на (рис. 34).

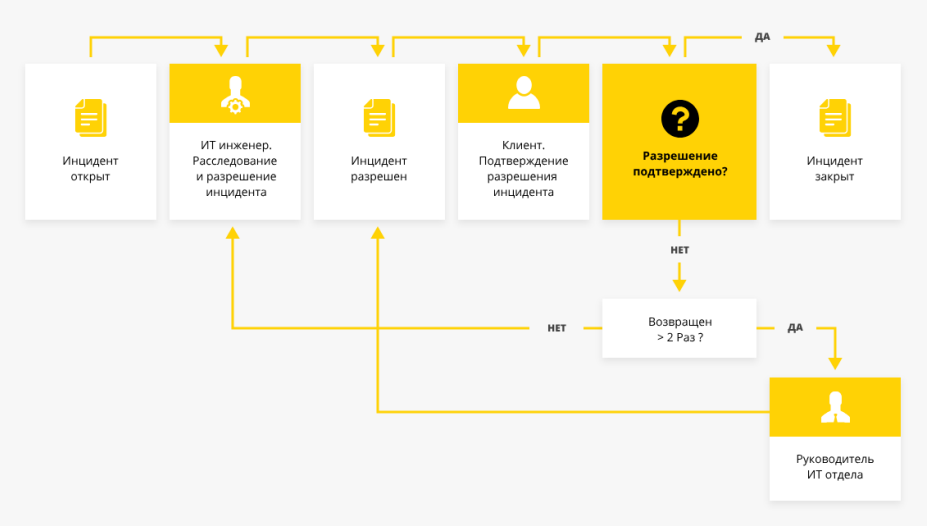


Рис. 34. Схема закрытия инцидента

Опишем основные преимущества повышения качества ИТ-обслуживания, которые получаются от внедрения нашей тикет-системы и механизмов описанных выше:

1. Фиксирование 100% обращений от клиента. Клиент может поставить задачу любым удобным способом, а также в любое время суток может обратится за помощью к сотруднику нашей поддержки. Поскольку система функционирует по принципу «единого окна», исключена ситуация когда клиент нам позвонил, а наш сотрудник сказал что вопрос не в его компетенции и кто его может решить он не знает или попросит перезвонить позднее.

2. Отслеживание всех инцидентов. Все инциденты фиксируются в системе и не теряются. В любой момент времени клиент может посмотреть статус всех своих инцидентов. В случае невозможности решить инцидент, происходит его эскалация. Также происходит информирование руководителя ИТ-отдела о затянутых и некачественно разрешенных инцидентах.

3. Гарантия качественного разрешения инцидента. Клиент обязательно подтверждает закрытие инцидента, таким образом исключена ситуация когда задачу клиента закрыли, а он остался недоволен.

4. Снижение количества критических инцидентов. Благодаря проактивному реагированию более половины критических инцидентов удается либо предотвратить заранее, либо среагировать на них быстрее, чем их обнаружит клиент.

Последовательность выполнения лабораторной работы

1. Придумать структуру небольшой компании.

2. Разработать блок-схему работы системы технической поддержки.

3. Придумать инцидент и описать последовательность работы сотрудников.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №8 «Выполнение обработки запросов в специализированной информационной системе»

Теоретические сведения

Рассмотрим процесс создания системы технической поддержки пользователей сайта недвижимости на базе системы Битрикс24.

Для начала работы необходимо развернуть у себя в организации саму систему Битрикс24 в какой-либо редакции, включить модуль CRM и поддержку бизнес-процессов. Далее нужно описать структуру компании (рис. 35).

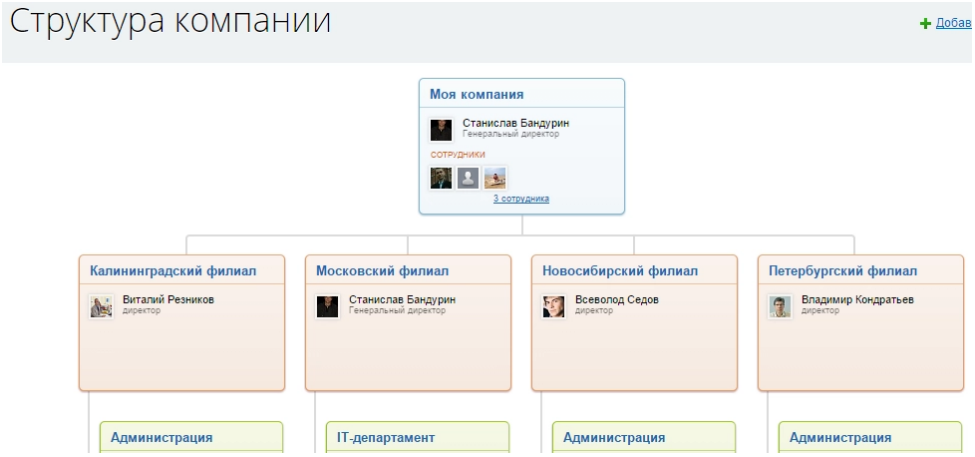


Рис. 35. Описание структуры компании в Битрикс24

Далее, нужно пригласить пользователей в настроенную штатную структуру организации в системе и в группы, определяющие функциональность сотрудников, которые в такие группы входят. Иными словами, группы должны соответствовать тем группам, которые описаны в бизнес-процессе.

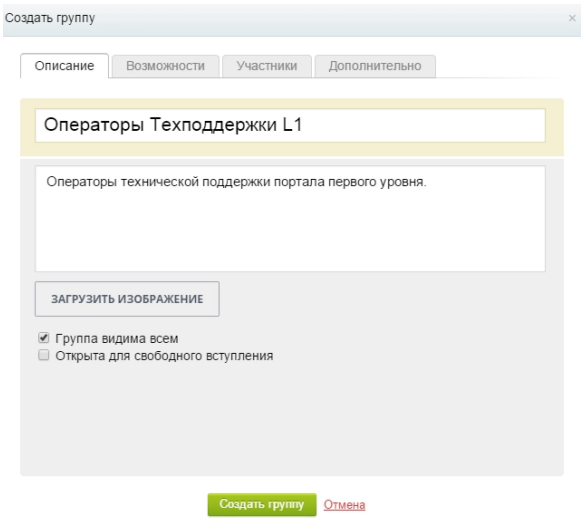


Рис. 36. Создание группы

Есть много вариантов имплементации бизнес-процессов с Битрикс24. Сразу же, что называется, «на берегу», необходимо определиться, в отношении какой сущности системы будут строиться бизнес-процессы. Поскольку данная тема (выбор, к какой сущности строить автоматизацию процессов) довольно обширна, я не буду останавливаться на ней в этой статье, а скажу только, что я выбрал сущность «Лид» CRM Битрикс24.

Таким образом, сущность «Лид» будет выполнять роль сущности «Обращение в службу техподдержки». В сущности «Лид» достаточно полей, которые мы можем настроить для того, чтобы он стал выполнять роль обращения. В качестве примера, изменю стандартное поле лида – «Источник» на необходимое для работы с обращением «Тип обращения».

А для того, чтобы поля списка соответствовали набору типов из Технического задания, отредактируем справочник «Источники» (который и исполняет роль справочника типов обращений), заполнив его необходимыми значениями (рис. 37).

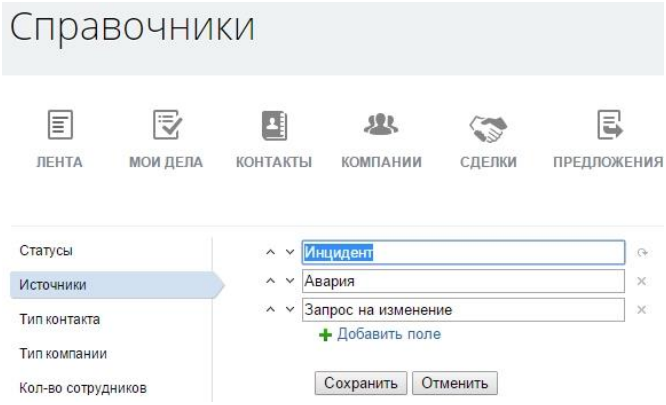


Рис. 37. Настройка справочника

Теперь применю один очень полезный функционал Битрикс24 – автоматическое создание Лидов из входящих писем электронной почты. Воспользуюсь настройками CRM Битрикс24, разделом «Битрикс24 Интеграция с почтой», где укажу адрес электронной почты, с которого необходимо принимать для генерации Лиды, которые будут давать начало бизнес-процессу обработки входящих обращений пользователей Интернет-портала. На этот адрес необходимо будет настроить сам интернет портал, чтобы он отправлял обращения именно туда (рис. 38).

По созданному Техническому Заданию с описанием бизнес-процессов, включая схемы, создаем рабочий процесс в системе для сущности «Лид». Этот процесс будет стартовать автоматически при добавлении Лида (который в том числе может быть создан пришедшим сообщением электронной почты) и обрабатываться операторами технической поддержки первого уровня.

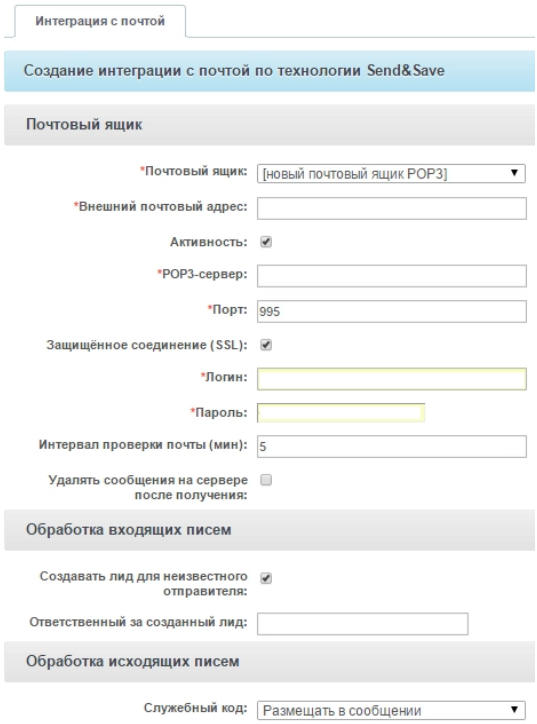


Рис. 38. Настройка почты

Последовательность выполнения лабораторной работы

1. Зарегистрировать бесплатный аккаунт в Битрикс24.

2. Заполнить справочники для тестовой компании, разработанной в предыдущей лабораторной работе.

3. Проверить обработку инцидента.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №9 «Решение и разбор примеров критических ситуаций в службе поддержки»

Теоретические сведения

Было бы здорово, если бы работа службы поддержки выглядела так. Клиент пишет в чат — менеджер за пару минут разбирается в ситуации — и получает благодарность. Все довольны и счастливы. Но в реальности клиентам может быть непонятно, тяжело, страшно — и они могут проявлять сильные эмоции. Порой это выливается в обилие сообщений, резкий тон и требования.

Справиться с эмоциями и решить проблему — задача службы поддержки. Даже в самых напряжённых ситуациях есть возможность договориться. Как минимум, сделать взаимодействие менее болезненным — и для клиента, и для компании.

Разберем несколько типовых критических ситуаций и методы реагирования на них.

1. Клиент не до конца понимает, как пользоваться продуктом

Описание ситуации. Не все клиенты — активные пользователи. Кто-то и вовсе запускает рассылку первый раз в жизни. Да и у постоянных клиентов могут возникать вопросы — это нормально. И даже после изучения всех материалов, которые обычно помогают, могут всплывать ошибки. Клиент теряется и не понимает, как ему быть — чувствуется сильный запрос в поддержке. Поэтому он может задавать больше вопросов, писать, что ничего не получается, и сердиться.

Что не поможет:

- Раздражаться. Ответы в духе «Смотрите в отправленном файле» точно сработают против. Клиент может закрыться и даже отказаться от услуг компании.
- Присылать стандартные инструкции. С большой вероятностью человек их уже читал и по каким-то причинам не понял. Нужен другой подход.

Что делать:

- Пройдите путь клиента вместе с ним по шагам. Проговорите, на какие кнопки нажимать, куда переходить, какой текст и куда вводить.
- В конце спросите, остались ли вопросы. Можно ещё узнать, что стало самым непонятным, чтобы эти сценарии больше не повторялись.

2. Клиент пока не доверяет компании

Описание ситуации. На клиентах в B2B лежит большая ответственность. Важно не просто выбрать подходящую платформу автоматизации маркетинга, но и настроить всё так, чтобы работало без сбоев. Поэтому не верить и перепроверять многие моменты — очень понятное поведение. Но иногда это выливается в сложности. Клиент может сомневаться в ответах поддержки, игнорировать её запросы или просить детальных описаний, как всё работает.

Что не поможет:

- Писать шаблонные ответы вроде «Ваше обращение важно для нас», «Мы делаем всё возможное, чтобы решить проблему». Это только раздражает — клиенту важно быть услышанным, а такие фразы выглядят как автоматические ответы.
- Поддаваться на возможные провокационные вопросы и писать эмоционально.

Что делать:

- За сильными эмоциями чаще всего скрывается страх потерять собственных клиентов или даже деньги. Вряд ли дело в «плохом» продукте, скорее в тревоге — и это полезно помнить.
- Отвечайте на все вопросы в спокойной форме. Если возможно, подкрепляйте сообщения доказательствами: отзывами других клиентов, скриншотами, случаями из практики, цифрами. Можно провести личную консультацию с клиентом.
- Если клиент всё же понёс какие-то потери, признайте свою ошибку. И дайте знать, что сделаете всё, чтобы избежать этого в будущем.

3. Клиент отправляет много рекомендаций.

Описание ситуации. Бывают случаи, когда клиенты приходят с проблемой, но при этом начинают критиковать продукт или работу специалистов. Это объяснимо — сложная ситуация могла отнять время или вызвать раздражение. Поэтому фразы клиентов могут казаться резкими, советов может быть очень много. Да, обратная связь ценна, но когда стоит задача срочно решить проблему — она отвлекает. От этого страдает даже не поддержка, а сами пользователи платформы.

Что не поможет:

- Спорить и защищаться. Это сильнее отвлечёт от решения проблемы.
- Отрицать абсолютно все рекомендации. Клиенты и правда могут подсветить что-то неочевидное, а это важно.

Что делать:

- Покажите клиентам, что их рекомендации имеют значение. Можно использовать нейтральные, но не шаблонные ответы: «Спасибо за рекомендацию, передам руководству», «Интересно! Мы это протестируем». Клиент должен почувствовать, что его мнение влияет на продукт. А если подобные запросы часто повторяются — тем более полезно прислушаться.

- Постарайтесь перевести внимание клиента на проблему. Объясните, что сначала решите её, а потом внимательно пройдётесь по рекомендациям.

- Если уже пробовали идею, которую рекомендуют, скажите об этом. Если клиент настаивает на каком-то решении, аргументированно объясните, почему это не ваш вариант, или ответьте, что вернётесь позже, как только проанализируете ситуацию.

4. Клиент просит решить проблему срочно.

Описание ситуации. При работе с платформой могут возникать сложности. И клиенты не всегда понимают, насколько критична та или иная ошибка. Иногда может казаться, что всё потеряно и уже ничего не исправить. Это сильнейший стресс. Поэтому они могут часто писать в чат и требовать скорее решить проблему.

Что не поможет:

- Отписываться стандартными фразами: «Ожидайте», «Мы решаем вашу проблему».

- Игнорировать. Даже если сообщения отправляют каждую минуту, важно показать, что вы всё видите.

Что делать:

- Постарайтесь прежде всего успокоить клиента. Объясните, что всё будет в порядке — работают профессионалы, они решают похожие проблемы каждый день.

- Скажите, сколько времени займёт решение вопроса. Если определить точное сложно, задайте диапазон: «Я сейчас всё проверю и вернусь в ближайшее время» Если проблема сложная, то можно обозначить срок «в течение дня».

- Если ситуация нештатная, а клиент просит скорее всё сделать — попробуйте по-человечески объяснить, почему вы не можете решить вопрос срочно. Можно сослаться на регламент, где прописаны сроки.

Или сказать, что у вас скопилась очередь из клиентов и вы стараетесь вникнуть в проблему каждого, чтобы все остались довольны.

5. Клиент переходит на эмоции, и причина непонятна.

Описание ситуации. Пожалуй, у каждой службы поддержки случались экстремальные ситуации. Когда клиент не мог объяснить, в чём проблема, и переходил на крайне сильные эмоции. В таком случае очень сложно определить, есть ли проблема — клиент может не отвечать или писать особенно жёстко.

Что не поможет:

- Спорить. Это вызовет ещё больше эмоций и не поможет продвинуться к решению.
- Грубить в ответ, даже если это желание кажется непреодолимым.

Что делать:

- За яркой эмоциональной реакцией вряд ли стоит конкретный сотрудник службы поддержки. Поэтому важно не принимать негатив на свой счёт. Дело может быть в очень разных причинах: тот же страх ошибиться, неудачный день, сложная ситуация в жизни. Но ещё это может быть реальная ошибка компании.
- Сначала постарайтесь понять, есть ли проблема на стороне компании. Задавайте спокойные заботливые вопросы в духе «Я хочу вам помочь. Какая у вас проблема?».
- Если проблема есть, старайтесь пропускать мимо эмоциональные фразы и идти по скрипту. Признайте эмоции клиента и покажите готовность решить проблему: «Да, вы столкнулись с ошибкой на платформе и имеете право злиться. Мы это обязательно исправим, но сейчас важно сосредоточиться на вопросе». Если проблема серьёзная, после её решения можно принести официальные извинения.
- Если клиент всё игнорирует и продолжает писать особенно эмоционально, выйдите из диалога. Объясните почему: «Мы не можем продолжить общение, потому что не понимаем, как помочь. Но если у вас появятся вопросы по работе платформы — обязательно ответим».

Как быть понятым в чате с клиентом: 7 важных правил.

При общении по телефону есть интонация, по которой проще понять, что имеет в виду клиент и какой у него настрой. Да и тот, кто звонит, может лучше понять сотрудника. Но в переписке интонаций нет — приходится полагаться только на текст.

Точнее донести мысль до клиента помогут несложные правила. Они написаны на основе опыта службы заботы о клиентах Sendsay, но вполне применимы и для других ниш.

1. Писать ёмко, избегать лишних слов.

2. Структурировать текст: разбивать на абзацы и делать списки.

3. Использовать повелительное наклонение, если это инструкция.

4. Расшифровывать специфические слова или давать ссылку на то, где о них можно почитать.

5. Задавать уточняющие вопросы в процессе. Перешёл ли клиент в нужное меню, увидел ли правильную плашку, та ли сумма ему высветилась.

6. Показывать всё, что можно показать. Особенно если речь о работе в интерфейсе. В идеале — использовать демонстрацию экрана. Как минимум — отправлять скриншоты.

7. В конце показать, что вы на связи.

Последовательность выполнения лабораторной работы

1. Придумайте и опишите модель поведения клиента.

2. Поработайте в паре для решения смоделированной критической ситуации.

3. Опишите ваши действия при разрешении критической ситуации.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №10 «Резервное копирование и восстановление файловой системы веб-браузера»

Теоретические сведения

Резервное копирование или бэкап браузера (от англ. "backup" – "резервная копия") – сохранение основных параметров браузера, таких как настройки, набор плагинов, закладки и пароли. Иногда параметры и расширения не сохраняют, а речь идёт лишь о сохранении закладок и паролей.

Бэкап браузера может потребоваться при переустановке Windows (или иной системы). Увы, о нём часто забывают и в итоге пользователю часто приходится заново по памяти восстанавливать все нужные ему параметры веб-обозревателя.

Общие способы бэкапа.

Сразу скажем, что за прошедшее время развития компьютерных технологий (и браузеров в частности) было разработано довольно много вариантов бэкапа. Сегодня каждый может выбрать себе тот, что придётся по вкусу именно ему. Все способы резервного копирования параметров браузеров можно разделить на три категории.

К первой категории относится встроенный в большинство современных веб-обозревателей механизм **автоматической синхронизации**. Всё, что нужно сделать, чтобы она работала – активировать **аккаунт** пользователя. С этого момента браузер сам будет сохранять резервные копии своих настроек на серверах разработчиков. Для восстановления всех закладок и паролей от нас потребуется лишь заново установить программу и войти в свою учётную запись. Единственный недостаток такого подхода в том, что мы полностью доверяем свою (в том числе и конфиденциальную) информацию третьим лицам.

Ко второй категории относится бэкап с использованием **стороннего софта** и/или специальных плагинов. Этот способ был весьма популярен в начале 2000-х годов, но сегодня в связи с изменениями в политиках распространения современных браузеров, практически не используется. Однако, если, например, у Вас стоит ещё старая Windows XP со старыми версиями Firefox или Google Chrome, то Вас вполне могут выручить программы, вроде MozBackup:

Наконец, к третьей категории относится полностью **ручное копирование** и восстановление файлов, хранящих те или иные настройки браузера. Данный способ является единственным возможным в том случае, если система не загружается, а к браузеру не было привязано никакого аккаунта. Чтобы реализовать его нужно знать версию браузера (чтобы

копируемые файлы подошли ко вновь установленному) и места, где лежат файлы с нужными данными. Этому способу мы и уделим максимум внимания (хотя, упомянём и про альтернативы).

Бэкап Хрома.

Google Chrome на сегодняшний день, несмотря на недовольство некоторых пользователей относительно последних обновлений, остаётся самым популярным браузером. Более 50% посетителей Интернета используют именно его, поэтому начнём с Хрома.

Проще всего всегда иметь резервную копию всех параметров браузера при помощи **аккаунта пользователя**. Чтобы активировать автоматический бэкап достаточно нажать на кнопку со стилизованным изображением человечка в правом верхнем углу (слева от кнопок свернуть/развернуть/закрыть) и авторизироваться при помощи своего аккаунта GMail:

После входа в аккаунт в "Настройках" Хрома Вы сможете задать, что именно нужно сохранять на серверах Гугла. Для этого найдите раздел "Вход" и там нажмите кнопку **"Дополнительные настройки синхронизации"**. По умолчанию активна опция "Синхронизировать все", но можно снять галочки с пунктов, которые Вам не нужны, выбрав опцию "Выбрать объекты для синхронизации" (рис. 39).

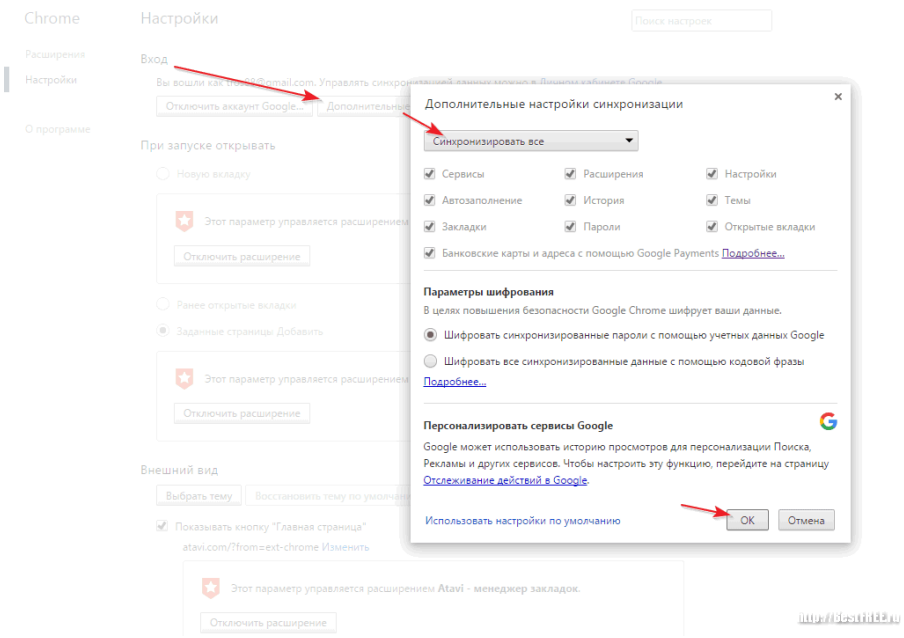


Рис. 39. Настройки бэкапа в Хроме

Самая большая сложность у Хрома с ручным сохранением параметров. Дело в том, что практически каждая новая версия изменяет структуру хранения личных данных пользователя, поэтому не исключено, что со временем описанное здесь перестанет соответствовать действительности.

Тем не менее, перечислю основные файлы, которые Вы можете сохранить и использовать в качестве бэкапа основного профиля (по умолчанию имеет название "Default", однако, если Вы делаете резервную копию другого профиля, то в адресах ниже вместо "Default" будет фигурировать имя копируемого профайла):

1. Закладки Хрома хранятся в файле с именем "Bookmarks" (без расширения) в папке C:\Users\ИМЯ_ПОЛЬЗОВАТЕЛЯ\AppData\Local\Google\Chrome\User Data\Default (или C:\Documents and Settings\ИМЯ_ПОЛЬЗОВАТЕЛЯ\Local Settings\Application Data\Google\Chrome\User Data\Default\Bookmarks для старых версий Windows).

2. Расширения обычно тоже находятся в папке профиля пользователя в специально выделенной папке с именем "Extensions": C:\Users\ИМЯ_ПОЛЬЗОВАТЕЛЯ\AppData\Local\Google\Chrome\User Data\Default\Extensions (или C:\Documents and Settings\ИМЯ_ПОЛЬЗОВАТЕЛЯ\Local Settings\Application Data\Google\Chrome\User Data\Default\Extensions). Кроме самой папки с расширениями могут быть дополнительные папки с параметрами этих расширений с именами "Extension Rules", "Extension State" и "Local Extension Settings". Лучше сохранить и их чтобы не терять настройки плагинов.

3. Пароли также хранятся в папке профиля пользователя в файле без расширения с именем "Login Data": C:\Users\ИМЯ_ПОЛЬЗОВАТЕЛЯ\AppData\Local\Google\Chrome\User Data\Default\ (или C:\Documents and Settings\ИМЯ_ПОЛЬЗОВАТЕЛЯ\Local Settings\Application Data\Google\Chrome\User Data\Default\).

4. Настройки Google Chrome находятся в файле "Preferences": C:\Users\ИМЯ_ПОЛЬЗОВАТЕЛЯ\AppData\Local\Google\Chrome\User Data\Default\ (или C:\Documents and Settings\ИМЯ_ПОЛЬЗОВАТЕЛЯ\Local Settings\Application Data\Google\Chrome\User Data\Default\). Однако, их не всегда удаётся потом восстановить, если версия нового браузера не совпадает с той, что была у Вас ранее.

Некоторые товарищи советуют для полного сохранения всех параметров Хрома полностью скопировать папку "Default" и затем заменить её во вновь установленном браузере. Однако, такой способ сработает лишь в том случае, если версии браузеров совпадают в точности до бет. Если же Вы попытаетесь заменить папку пользователя в более новой версии Google Chrome, то можете получить ошибку или вообще полный крах веб-

обозревателя. Поэтому лучше всё-таки восстанавливать параметры "точечно".

Бэкап Файрфокса.

Третьим в списке лидеров среди браузеров всегда был и остаётся Mozilla Firefox (правда, он практически сровнялся по количеству приверженцев с Opera, поэтому по некоторым данным, возможно, является и вторым после Хрома).

В новых версиях Файрфокс всё больше старается походить на Хром, перенимая у него многие возможности. Среди прочего появился и собственный механизм онлайн-синхронизации. Чтобы активировать его нужно вызвать меню в правом верхнем углу и нажать на кнопку "Войти в Синхронизацию" или перейти в раздел "Настройки" – "Синхронизация" (рис. 40).

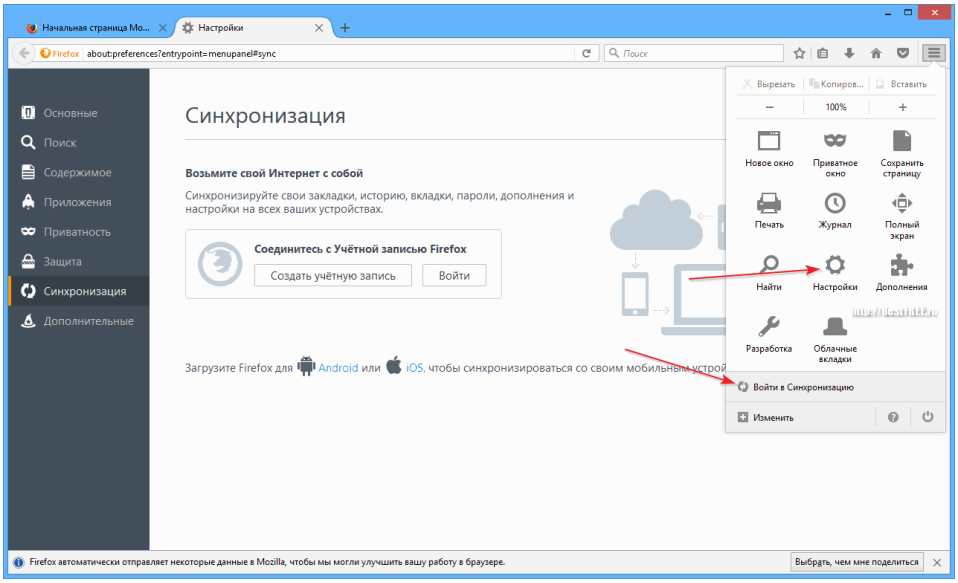


Рис. 40. Настройки бэкапа Файрфокс

Используя сервис синхронизации от Mozilla Вы не только сохраните все параметры браузера, но также сможете синхронизировать закладки и пароли между десктопной и мобильной версией веб-обозревателя (впрочем, в Хроме это тоже работает).

Что касается программ для бэкапа Firefox, то здесь поможет уже упомянутый выше MozBackup.

Несмотря на то, что программа уже давно не обновлялась, она до сих пор в состоянии делать нормальные бэкапы даже самых последних версий Файрфокса и почтового клиента Thunderbird (это так, между прочим :)).

Если же Вы решите делать бэкап вручную, то Вам потребуется сохранить несколько файлов по адресу C:\Users\ИМЯ_ПОЛЬЗОВАТЕЛЯ\AppData\Roaming\Mozilla\Firefox\Profiles\

(или C:\Documents and Settings\ИМЯ_ПОЛЬЗОВАТЕЛЯ\Local Settings\Application Data\Mozilla\Firefox\Profiles для старых версий Windows). Как и в случае с Хромом, хранится они будут, скорее всего, в папке под названием "Default" (либо прямо в папке Profiles). Если же у Вас несколько профилей, то вместо "Default" ищите в указанных путях директорию с именем своего профиля:

1. Закладки Mozilla Firefox, а заодно с ними история загрузок и просмотров страниц, хранятся в файле "places.sqlite".
2. Пароли содержатся сразу в двух файлах "key3.db" (сам список палолей) и "logins.json" (параметры автозаполнения форм авторизации). Кстати, за работу автозаполнения форм, в которых не используется авторизация, отвечает файл "formhistory.sqlite", который тоже можно сохранить.
3. Расширения (если они установлены) хранятся в отдельной папке Вашего профиля под названием "extensions".
4. Настройки тоже находятся в нескольких файлах: общие в "prefs.js", пользовательские в "user.js", а параметры панелей инструментов в "xulstore.json".
5. Настройки сайтов, например, разрешения на запуск плагинов или параметры отображения, находятся в файлах "permissions.sqlite" и "content-prefs.sqlite".

Зная где и что искать, сохранить настройки Вашего любимого браузера не составит особого труда, даже, если Ваша операционная система вдруг перестанет работать. Просто скопируйте нужные файлы и замените ими впоследствии данные в новоустановленном браузере. Если же Вы не хотите возиться с бэкапами вручную, то Вам пригодятся встроенные во всех современных веб-обозревателях функции автоматической синхронизации. Достаточно раз создать активировать аккаунт и программа сама будет сохранять резервные копии всех настроек на серверах разработчиков.

Последовательность выполнения лабораторной работы

1. Сделайте резервную копию Chrome.

2. Сделайте резервную копию Firefox.

3. Восстановите резервные копии Chrome и Firefox на другом компьютере.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №11 «Резервное копирование и восстановление базы данных веб-приложения»

Теоретические сведения

В современном мире, управляемом данными, невозможно переоценить важность наличия мощной стратегии резервного копирования и восстановления данных. Последовательный рост объема данных и решающая роль данных в бизнес-операциях требуют от организаций необходимости защищать свою ценную информацию.

Стратегии резервного копирования и восстановления данных обеспечивают согласованность данных, предотвращают потерю данных, поддерживают доверие заинтересованных сторон и обеспечивают непрерывность бизнеса. Хорошо продуманная стратегия резервного копирования и восстановления данных имеет решающее значение для защиты от различных рисков, таких как повреждение данных, сбои оборудования, нарушения безопасности и человеческие ошибки. Имея надежный план резервного копирования и восстановления, организации могут минимизировать влияние непредвиденных случаев потери данных, эффективно восстанавливать свои данные и сокращать время простоев.

Более того, комплексная стратегия резервного копирования и восстановления позволяет организациям соблюдать нормативные требования и отраслевые стандарты в отношении защиты данных. Обеспечение безопасности и постоянной доступности конфиденциальных данных имеет решающее значение для поддержания доверия клиентов, выполнения юридических обязательств и предотвращения финансовых и репутационных потерь.

Понимание различных типов резервных копий баз данных.

При разработке стратегии резервного копирования базы данных важно учитывать различные типы доступных резервных копий. Понимание различных типов резервного копирования помогает обеспечить адекватную защиту ваших данных и достичь желаемой эффективности восстановления. К основным типам резервных копий баз данных относятся:

Полные резервные копии.

Полная резервная копия — это комплексная резервная копия всей базы данных, включая все файлы данных, индексы и другие компоненты. Полные резервные копии считаются наиболее надежной формой резервного копирования, поскольку они хранят полную копию вашей базы данных, что упрощает восстановление в случае потери данных. Тем не менее, полное резервное копирование может занять много времени и ресурсов, в зависимости от размера вашей базы данных.

Инкрементальное резервное копирование.

В добавочных резервных копиях хранятся только те данные, которые изменились с момента последнего полного или добавочного резервного копирования. Это делает их более эффективными по времени и пространству, чем полные резервные копии, поскольку они сохраняют только изменения, внесенные в базу данных между резервными копиями. Процесс восстановления инкрементальных резервных копий может быть более сложным: вам потребуется восстановить полную резервную копию и применить последующие инкрементные резервные копии в правильном порядке.

Дифференциальное резервное копирование.

Дифференциальные резервные копии сохраняют данные, которые изменились с момента последнего полного резервного копирования. Это означает, что размер дифференциальных резервных копий со временем увеличивается, поскольку накапливается больше изменений, но они остаются меньше, чем полная резервная копия. Когда дело доходит до восстановления данных, дифференциальное резервное копирование предлагает более простой подход, чем инкрементальное резервное копирование, поскольку для восстановления базы данных вам нужна только последняя полная резервная копия и последняя дифференциальная резервная копия.

При разработке стратегии резервного копирования базы данных необходимо учитывать несколько факторов, чтобы обеспечить адекватную защиту ваших данных и соответствие процесса восстановления требованиям вашей организации. Вот некоторые важные факторы, которые следует учитывать:

Тип базы данных.

Различные системы баз данных имеют особые требования и возможности для резервного копирования и восстановления данных. Обязательно изучите и поймите встроенные параметры резервного копирования и восстановления, доступные для выбранной вами системы базы данных.

Частота резервного копирования.

Определите, как часто следует выполнять резервное копирование, исходя из важности ваших данных, скорости изменения данных и устойчивости вашей организации к рискам. Установление подходящей частоты резервного копирования гарантирует достижение целевых показателей точки восстановления (RPO) и минимизацию последствий потери данных.

Место хранения резервной копии.

Надежное хранение резервных копий имеет жизненно важное значение для обеспечения безопасности и доступности данных. Распространенной практикой является правило резервного копирования 3-2-1, которое предполагает хранение трех копий ваших данных на двух разных типах носителей, причем по крайней мере одна копия хранится вне офиса.

Целевое время восстановления (RTO).

Целевое время восстановления означает количество времени, которое вы можете принять, прежде чем система будет восстановлена после инцидента с потерей данных. Понимание вашего RTO помогает спланировать процесс восстановления и принять соответствующие решения относительно процессов резервного копирования и восстановления.

Целевые точки восстановления (RPO).

Цели точки восстановления определяют максимальный объем данных, который ваша организация может позволить себе потерять в случае сбоя. Определите свою RPO, оценив потенциальное влияние потери данных на бизнес-операции. Установление RPO помогает определить частоту и тип резервного копирования, необходимые для вашей базы данных. Тщательно учитывая эти факторы, вы можете разработать эффективную стратегию резервного копирования и восстановления, отвечающую конкретным потребностям вашей организации и снижающую риски, связанные с потерей данных.

Популярные инструменты для резервного копирования и восстановления баз данных.

Доступно несколько инструментов для эффективного управления резервным копированием и восстановлением базы данных. Некоторые из этих инструментов являются встроенными в систему управления базами данных (СУБД) , а другие представляют собой сторонние решения или облачные сервисы. Давайте рассмотрим несколько популярных вариантов:

Собственные утилиты СУБД.

Большинство решений СУБД, таких как MySQL, PostgreSQL , Oracle и Microsoft SQL Server, предоставляют встроенные утилиты и команды для управления резервным копированием и восстановлением. Эти инструменты, как правило, надежны и адаптированы для бесперебойной работы с соответствующими базами данных. Команды и утилиты могут различаться в разных системах баз данных, но некоторые общие инструменты включают в себя:

- `mysqldump` и `mysqlbackup` для MySQL;
- `pg_dump` , `pg_basebackup` и `pg_restore` для PostgreSQL;

- `RMAN` (менеджер восстановления) для Oracle;
- `SQL Server Management Studio` (SSMS) и команды `Backup` / `Restore` для Microsoft SQL Server.

Сторонние решения.

Некоторые организации предпочитают сторонние решения для управления резервным копированием и восстановлением баз данных. Эти инструменты часто поддерживают несколько систем баз данных и предлагают дополнительные функции, такие как улучшенная автоматизация, централизованное управление и расширенное шифрование. Некоторые популярные сторонние инструменты включают в себя:

- Veeam: предоставляет быстрые и надежные решения для резервного копирования, восстановления и репликации для виртуальных, физических и облачных сред.
- Acronis: предлагает набор решений для резервного копирования, аварийного восстановления и безопасного доступа к данным, включая поддержку популярных баз данных.
- Рубрик: предоставляет решение для управления облачными данными, которое интегрируется с различными базами данных для автоматизации резервного копирования, восстановления и тестирования.

Облачные сервисы.

Облачные службы баз данных, такие как Amazon RDS, база данных SQL Microsoft Azure и Google Cloud SQL, предоставляют встроенные функции для управления резервным копированием и восстановлением. Эти услуги обычно включают восстановление на определенный момент времени и автоматическое резервное копирование моментальных снимков, что упрощает процесс резервного копирования и восстановления. Эти услуги также могут обеспечивать географическую избыточность, что улучшает планирование аварийного восстановления:

- Amazon RDS: предлагает управляемую службу реляционной базы данных, которая поддерживает различные ядра баз данных, включая MySQL, PostgreSQL, Oracle, SQL Server и Amazon Aurora.
- База данных SQL Microsoft Azure: предоставляет полностью управляемую службу для баз данных SQL Server, позволяя пользователям сосредоточиться на оптимизации производительности и сокращении затрат, а не на управлении резервным копированием и других задачах инфраструктуры.

- Google Cloud SQL: предоставляет полностью управляемую службу реляционных баз данных для MySQL, PostgreSQL и SQL Server, предлагая функции автоматического резервного копирования и восстановления, а также высокую доступность и региональную репликацию.

Последовательность выполнения лабораторной работы

1. В зависимости от особенностей вашего веб-сайта выберите оптимальную стратегию резервного компирования.

2. Опишите выбранный алгоритм резервного копирования.

3. Показать результаты работы преподавателю.

4. Ответить на вопросы преподавателя.

Лабораторная работа №12 «Использование сценариев и скриптов для организации процесса резервирования и восстановления данных»

Теоретические сведения

В мире, где цифровые данные играют глапную роль, обеспечение их безопасности и целостности становится все более важным. Особенно важно иметь систему резервного копирования (бэкапа), которая не только сохраняет данные, но и делает это умно и эффективно. В этой статье мы узнаем процесс создания автоматизированной системы бэкапа с использованием скриптов, обеспечивая надежную защиту ваших ценных данных.

Основные принципы бэкапа.

Прежде чем приступить к созданию автоматизированной системы бэкапа, важно понять основные принципы её работы. Вот некоторые из них:

1. Регулярность: бэкапы должны создаваться регулярно, чтобы минимизировать потерю данных в случае сбоя.
2. Инкрементальность: инкрементальные бэкапы сохраняют только измененные или добавленные файлы, что экономит место на носителе.
3. Хранение: бэкапы должны храниться в безопасном месте, защищенном от физических и киберугроз.
4. Проверка целостности: после создания бэкапа необходимо проверить его целостность, чтобы убедиться, что данные были успешно скопированы.

Использование скриптов для автоматизации.

Перед началом создания скрипта для автоматизации бэкапа необходимо выбрать язык программирования. Распространенный выбор для администрирования Linux-серверов это Bash, но также можно использовать Python или Perl для более сложных сценариев.

Настройка параметров бэкапа.

Перед написанием скрипта определите параметры бэкапа: какие данные будут копироваться, куда они будут сохраняться, как часто будет выполняться резервное копирование и сколько храниться бэкапов.

Создание скрипта бэкапа.

Пример на Bash для создания инкрементального бэкапа базы данных MySQL:

```
bash
```

```
#!/bin/bash

# Параметры базы данных

DB_USER="username"

DB_PASS="password"

DB_NAME="database_name"

# Директория для хранения бэкапов

BACKUP_DIR="/backup"

# Создание директории, если она не существует

mkdir -p $BACKUP_DIR

# Имя файла для бэкапа

BACKUP_FILE="$BACKUP_DIR/db_backup_$(date +%Y-%m-%d).sql.gz"

# Создание инкрементального бэкапа

mysqldump -u $DB_USER -p$DB_PASS $DB_NAME | gzip > $BACKUP_FILE

# Удаление старых бэкапов (если требуется)

# find $BACKUP_DIR -type f -name "*.sql.gz" -mtime +7 -delete
```

Этот скрипт создает инкрементальный бэкап базы данных MySQL и сохраняет его в указанной директории.

Настройка планировщика задач.

Чтобы скрипт выполнялся автоматически, настройте планировщик задач cron в Linux, чтобы он запускал скрипт с заданной периодичностью.

Мониторинг и уведомления.

После настройки автоматизированной системы бэкапа на скриптах нужно реализовать мониторинг процесса и уведомления об ошибках или проблемах. Это даст быстро реагировать на любые сбои или неполадки в создании бэкапов и своевременно принимать меры по их устранению.

Использование утилит мониторинга.

Существует много утилит для мониторинга, например Nagios, Zabbix, Prometheus и другие, которые дают отслеживать состояние системы и

процессов, включая выполнение скриптов бэкапа. Подключение таких утилит к системе даст быстро получать уведомления о проблемах и сбоях.

Настройка уведомлений.

После выбора утилиты мониторинга необходимо настроить уведомления о событиях, связанных с процессом бэкапа. Это может быть отправка электронных писем, SMS-сообщений, уведомлений в мессенджерах или интеграция с системами управления инцидентами (Incident Management Systems).

Пример скрипта для отправки уведомлений:

```
bash

#!/bin/bash

# Параметры для отправки уведомлений

EMAIL="admin@example.com"

MESSAGE="Произошла ошибка при выполнении бэкапа. Пожалуйста, проверьте систему."

# Отправка уведомления по электронной почте

echo "$MESSAGE" | mail -s "Ошибка бэкапа" $EMAIL
```

Этот скрипт отправляет уведомление на заданный адрес электронной почты в случае ошибки при выполнении бэкапа.

Регулярное тестирование восстановления.

Наконец, важным шагом в обеспечении эффективности автоматизированной системы бэкапа это регулярное тестирование процесса восстановления данных. Это дает убедиться, что созданные бэкапы действительно могут быть успешно восстановлены при необходимости.

План тестирования восстановления.

- Регулярно проводите тестирование восстановления на тестовом сервере или в виртуальной среде.
- Оцените время восстановления данных и проверьте их целостность.
- Запишите и документируйте процедуру восстановления для оперативной реакции в случае реального инцидента.

Автоматизация тестирования восстановления.

Можно также автоматизировать процесс тестирования восстановления, создав специальные скрипты или сценарии, которые будут автоматически восстанавливать данные из бэкапа и проверять их целостность.

Резервное копирование файловой системы Linux.

В системе Linux регулярное резервное копирование имеет решающее значение для защиты ваших данных, обеспечения их сохранности и упрощения восстановления в случае сбоя. Независимо от того, работаете ли вы с личными файлами, настраиваете среду разработки или управляете серверами, вам нужен надёжный план резервного копирования.

Один из самых простых и гибких способов резервного копирования — написание собственного сценария, который позволяет вам решать, что именно будет копироваться, когда и где это будет храниться. Это также экономит время и снижает риск потери данных.

В этой статье вы узнаете, как создать простой сценарий оболочки, который автоматически создаёт резервные копии важных файлов.

Зачем использовать сценарий резервного копирования?

Использование сценария резервного копирования в Linux даёт системным администраторам больше возможностей и гибкости при защите важных данных.

Вот почему это разумный ход:

- Автоматическое резервное копирование: вы можете запланировать выполнение скрипта с помощью таких инструментов, как cron, чтобы вам не приходилось вручную создавать резервные копии.
- Полный контроль: вы сами решаете, что создавать в резервных копиях, куда их сохранять и как часто. Будь то ежедневные копии `/etc`, еженедельные снимки домашних каталогов или просто выборочные каталоги.

Создание сценария резервного копирования важных файлов.

Откройте терминал и перейдите в каталог, в котором вы хотите сохранить скрипт (например, в ваш домашний каталог или `/usr/local/bin/` для доступа из всей системы).

Создайте файл сценария с помощью текстового редактора, например nano или vim:

```
nano backup_script.sh
```

Ниже приведен базовый скрипт, который создает резервные копии файлов в указанный каталог.

```
#!/bin/bash


# Определите исходный и целевой каталоги

SOURCE_DIR="/home/user/Documents"     #     Исходный каталог для резервного копирования

BACKUP_DIR="/home/user/backups"  #  Целевой  каталог для резервных копий


# Создайте метку времени для папки с резервными копиями

TIMESTAMP=$(date +'%Y%m%d%H%M%S')


# Создайте новую папку с резервными копиями с меткой времени

BACKUP_FOLDER="$BACKUP_DIR/backup_$TIMESTAMP"

mkdir -p "$BACKUP_FOLDER"


# Скопировать файлы в папку с резервными копиями

cp -r "$SOURCE_DIR"/* "$BACKUP_FOLDER"


# Записать в журнал завершение резервного копирования

echo "Резервное копирование завершено в $TIMESTAMP" >> "$BACKUP_DIR/backup_log.txt"


# Необязательно: удалите резервные копии старше 30 дней

find "$BACKUP_DIR" -type d -name "backup_*" -mtime +30 -exec rm -rf {} \;
```

Объяснение сценария:

- `SOURCE_DIR`: Каталог, содержащий файлы, которые вы хотите создать резервную копию (например, `/home/user/Documents`).
- `BACKUP_DIR`: Каталог, в котором будут храниться резервные копии (например, `/home/user/backups`).
- `TIMESTAMP`Переменная, которая хранит текущую дату и время для уникальной идентификации каждой папки с резервными копиями.
- `mkdir -p`: Создает папку резервной копии с меткой времени.
- `cp -r`: Копирует все файлы и подкаталоги из исходного каталога в папку с резервными копиями.
- Команда echo регистрирует завершение резервного копирования и добавляет метку времени в файл журнала.
- Команда find удаляет резервные копии старше 30 дней, чтобы каталог резервных копий не заполнялся старыми файлами.

Сохраните и закройте скрипт (в nano нажмите `CTRL + X`, затем `Y` для подтверждения и нажмите Enter).

Теперь, когда скрипт написан, вам нужно сделать его исполняемым.

```
chmod +x backup_script.sh
```

Теперь вы можете запустить скрипт вручную, выполнив:

```
./backup_script.sh
```

Если всё настроено правильно, он должен создать резервную копию ваших файлов в указанном каталоге для резервного копирования.

Запланируйте сценарий резервного копирования С помощью Cron.

Чтобы автоматизировать процесс резервного копирования, вы можете запланировать выполнение сценария через определённые промежутки времени с помощью `cron`, планировщика заданий Linux.

Откройте файл конфигурации cron, набрав:

```
crontab -e
```

Добавьте задание cron для запуска скрипта в указанное время. Например, чтобы запускать скрипт резервного копирования каждый день в 2 часа ночи, добавьте в файл cron следующую строку:

```
0 2 * * * /path/to/backup_script.sh
```

Теперь скрипт будет запускаться автоматически в указанное время.

Чтобы подтвердить создание резервной копии, вы можете проверить каталог резервных копий и убедиться, что резервные копии создаются должным образом, или просмотреть журналы cron, чтобы убедиться, что сценарий резервного копирования запускается в запланированное время.

```
grep CRON /var/log/syslog
```

Добавьте сжатие с помощью tar и gzip

Вместо того чтобы копировать файлы напрямую, вы можете сжать всю резервную копию в архив `.tar.gz` для экономии места и поддержания порядка.

Модифицированный скрипт со сжатием:

```
# Определите исходный и целевой каталоги
SOURCE_DIR="/home/user/Documents"
BACKUP_DIR="/home/user/backups"

# Создайте метку времени
TIMESTAMP=$(date +'%Y%m%d%H%M%S')

# Определите имя файла резервной копии
BACKUP_FILE="$BACKUP_DIR/backup_$TIMESTAMP.tar.gz"

# Создайте каталог резервных копий, если он не существует
mkdir -p "$BACKUP_DIR"
```

```
# Создайте сжатый архив исходного каталога
tar -czf "$BACKUP_FILE" -C "$SOURCE_DIR" .

# Запишите в журнал завершение резервного копирования
echo "Сжатая резервная копия создана в $TIMESTAMP" >> "$BACKUP_DIR/backup_log.txt"

# Необязательно: удалите старые резервные копии (более 30 дней назад)
find "$BACKUP_DIR" -type f -name "backup_*.tar.gz" -mtime +30 -exec rm -f {} \;
```

Отправка резервной копии на удаленный сервер с помощью scp.

Если вы хотите хранить резервные копии на удалённом сервере (например, на резервном VPS или NAS), вы можете загрузить архив с помощью команды scp.

Добавьте следующее в конец скрипта:

```
# Задайте информацию об удалённом сервере
REMOTE_USER="ваш_пользователь"
REMOTE_HOST="ваш_сервер_ip"
REMOTE_DIR="/удаленный/резервная_копия/местоположение"

# Отправьте файл резервной копии на удалённый сервер
scp "$BACKUP_FILE" "$REMOTE_USER@$REMOTE_HOST:$REMOTE_DIR"
```

Настройте аутентификацию по ключу SSH между вашим локальным компьютером и удалённым сервером, чтобы не вводить пароли вручную.

Последовательность выполнения лабораторной работы

1. Используйте приведенные выше скрипты для проверки резервного копирования вашего сайта.

2. Адаптируйте данные скрипты под структуру вашего сайта.

3. Показать результаты работы преподавателю.

4. Ответить на вопросы преподавателя.

Лабораторная работа №13 «Настройка прав доступа к файловой системе и базе данных»

Теоретические сведения

Контроль доступа в Windows Server с помощью NTFS — это основа защиты данных: настройка прав по ролям, аудит событий, предотвращение утечек и защита от несанкционированного доступа. Особенно это важно при работе на VPS, где безопасность данных напрямую влияет на стабильность и доверие к вашему проекту.

Представьте, что в одной папке хранятся зарплатные отчёты, сканы паспортов сотрудников, внутренние чертежи новых продуктов и договоры с ключевыми клиентами. Если все пользователи получат к ней одинаковый доступ — риски очевидны. Бухгалтеру незачем видеть черновики отдела R&D, а стажёру — знакомиться с кадровыми документами.

Каждый должен видеть только то, что необходимо для выполнения своих задач. Более того, права должны различаться: кто-то получает доступ только на чтение, кто-то может редактировать, а кто-то не видит папку вовсе.

Система NTFS в Windows Server позволяет точно задать такие параметры. А если подключить аудит безопасности, можно отследить, кто и когда открывал нужный документ, а также были ли попытки изменения или удаления. Это особенно важно для расследования инцидентов и защиты от случайных ошибок.

Чтобы настроить права доступа, включите сервер и войдите в систему под учётной записью администратора. Откройте «Проводник» и перейдите к нужной папке, например:

```
D:\Документы\Бухгалтерия
```

Щёлкните правой кнопкой мыши по папке → выберите «Свойства» → вкладка «Безопасность».

Именно файловая система NTFS (New Technology File System) — стандартная для Windows Server — позволяет гибко управлять правами: чтение, изменение, удаление, создание файлов. Причём это можно делать как для отдельных пользователей, так и для групп.

На вкладке «Безопасность» нажимаем «Изменить», чтобы назначить или ограничить доступ.

Дальше — «Добавить», вводим имя пользователя или группы (например, Бухгалтерия).

Потом — «Проверить имена», чтобы система подтвердила, что такой пользователь есть.

Указываем, какие права выдаём:

- чтение — человек может только открыть и просмотреть файл;
- изменение — можно редактировать и сохранять;
- полный доступ — всё, включая удаление и смену прав.

Основа этой модели доступа — каждый получает только те права, которые нужны ему для работы.

Построение ролевой модели прав: создаём группы и управляем доступом.

Чтобы не настраивать права доступа для каждого сотрудника вручную, используется ролевая модель доступа. Суть простая: доступ не лично кому-то, а группе «Бухгалтерия», например. Когда человек уходит или приходит, ничего на сервере менять не надо — просто добавляете или удаляете его из группы.

Как это сделать на практике? Открываем управление пользователями и группами. На сервере нажмите Пуск, введите Computer Management (или «Управление компьютером») и откройте его.

Слева выберите Local Users and Groups → Groups (или «Локальные пользователи и группы» → «Группы»).

Создаём нужные группы:

- правый клик по папке Groups → New Group;
- вводим имя, например: Отдел_Бухгалтерии, Юристы, ИТ;
- Нажмите Create («Создать»).

Добавляем пользователей в группу:

- выберите и нажмите на нужную группу → Add… → имя пользователя или выберите его из списка → ОК.

Готово — теперь сотрудник унаследует все права, которые связаны с этой группой.

Настраиваем права для групп на нужные папки.

Откройте свойства папки (например, с файлами бухгалтерии). Перейдите на вкладку Безопасность → нажмите «Изменить». Затем «Добавить» и название группы («Отдел_Бухгалтерии») → ОК.

Установите нужные права (например, Чтение, Изменение, Полный доступ). Подтвердите изменения — готово.

Это упрощает управление правами, снижает риск ошибок, делает контроль в Windows Server понятным и удобным даже для тех, кто впервые это настраивает.

Настройка аудита NTFS: кто, когда и с чем работал.

Если хотите узнать, кто открывал, менял или удалял важные файлы, нужно включить аудит доступа. Тогда все действия будут записываться в журнал событий, и в любой момент можно посмотреть, что происходило с файлами.

Открываем политику безопасности. Нажмите Win+R, введите secpol.msc и — Enter.
Откроется окно Локальная политика безопасности (может называться «Local Security Policy»).

Если появляется сообщение об ошибке — возможно, вы используете редакцию Windows Home, в которой эта функция недоступна.

Включаем аудит доступа. В левой части экрана откройте:

Политики локальной безопасности → Локальные политики → Политика аудита.

В правой части найдите пункт «Аудит доступа к объектам», дважды кликните.

Установите галочки на:

- успешные;
- неудачные.

Нажмите ОК для активации.

Указываем, что именно отслеживать, например в: D:\Документы\Бухгалтерия:

- правой кнопкой по папке, затем Свойства → Безопасность → кнопка Дополнительно;
- в разделе Аудит нажать Добавить.

Настраиваем, что отслеживать. Введите имя пользователя или группы, действия которых хотите отслеживать (например, Все — для всех).

Отметьте нужное: чтение, изменение, удаление.

Нажмите ОК и закройте все окна.

Когда кто-то будет открывать, менять или удалять файл — информация об этом появится в Журнале событий Windows:

- откройте Пуск → введите Просмотр событий → откройте приложение;
- перейдите в: Журналы Windows → Безопасность.

Здесь будут отображаться события с кодами (чтение/изменение файлов):

- 4663 — попытка доступа к файлу;
- 4656 — проверка прав перед доступом.

Настройка прав доступа в Linux.

Операционные системы на базе Linux являются многопользовательскими, и поэтому вопрос разграничения доступа к файлам и директориям является важным и требующим внимания.

Механизм разграничения доступа базируется на именах пользователей, а также на названиях групп пользователей, состав которых изменяет и определяет root пользователь (суперпользователь). Пользователь может входить в одну или несколько групп и иметь права доступа в зависимости от группы, в которую он входит.

При создании файла у него появляется владелец, то есть пользователь, который запустил процесс его создания. Также определяется группа, которая будет иметь права на этот файл. Изменять владельца файла и группу файла можно при помощи команд chown и chgrp.

Команды chown и chgrp.

Команда chown (расшифровывается как “change owner”) используется для изменения владельца файла. При этом выполнять изменение владельца должен обязательно суперпользователь. Вам нужно ввести название команды, затем имя пользователя, которого вы собираетесь сделать владельцем файла, и в конце название файла:

```
# chown имя_пользователя название_файла
```

Команда chgrp (сокращение от “change group”) используется для изменения группы файла. При этом выполнить команду может как суперпользователь, так и владелец файла, но он обязательно должен быть членом группы, которой он хочет передать права на файл. Вам нужно ввести команду, название группы файлов, которой вы передаете права, а затем название файла:

```
# chgrp название_группы название_файла
```

Сразу расскажем о полезном ключе -R (расшифровывается как “recursively”). Он позволяет применять команду не только к текущей директории, но и ко всем поддиректориям. Ключ можно использовать и с chown, и с chgrp. Использовать такую рекурсивную команду удобно в случае большой вложенности.

Права доступа.

Права доступа всегда необходимо назначать и разграничивать – это крайне важный момент обеспечения безопасности вашей Linux-системы. В случае, когда хакеру удастся получить доступ к одному из ваших пользователей, грамотно настроенные права доступа к файлам и каталогам не дадут ему возможности сделать много неприятностей. Иными словами, настройка прав доступа даст вам возможность максимально ограничить ваши данные от попадания в чужие руки (естественно, если речь не идет о root или sudo пользователях без ограничений, которые могут изменять любые права под себя).

Перед тем, как перейти к самому процессу изменения прав, нужно понять, как смотреть уже имеющиеся права у файлов и каталогов. Делать это можно при помощи команды ls и ключа –l.

```
# ls -l
```

После ввода этой команды вы увидите содержимое текущего каталога, а также информацию о владельце и правах доступа.

Пример отобразившейся информации:

```
total 32
drwxr-xr-- 2 root root 4096 May 7 2016 bin
drwxr-xr-- 2 root root 4096 May 7 2016 etc
drwxr-xr-- 2 root root 4096 Aug 11 2016 games
drwxr-xr-- 2 user1 users May 5 12:00 Images
drwxr-xr-- 2 root root 4096 Mar 23 2016 sbin
-rw-r--r-- 1 root root May 6 20:28 file1.txt
-rw-r--r-- 1 root root May 6 09:58 kdm.log
```

Разберемся, что означают эти строки.

Первая буква в выводе обозначает тип файла. Самые популярные обозначения, которые вы будете встречать чаще всего, это:

- - — обычный файл;

- d – каталог.

Помимо них есть и другие обозначения:

- b — файл блочного устройства;
- c — файл символьного устройства;
- s — доменное гнездо (socket);
- p — именованный канал (pipe);
- l — символическая ссылка (link).

Три следующие буквы, которые идут после первой, означают те права доступа, которые имеет пользователь-владелец этого файла или каталога. Расшифровываются они следующим образом:

- r – read – право на чтение;
- w – write – право на запись (изменение, в том числе и удаление);
- x – execute – право на выполнение этого файла (если речь о каталоге, то просмотр оглавления и поиск в нем);
- - (прочерк) вместо одной из букв говорит о том, что соответствующего права у вас нет.

Поэтому по записи

-rw-r–-r-- 1 root root May 6 20:28 file1.txt

можно сказать о том, что это обычный файл, владельцем которого является пользователь root, и он может читать и изменять этот файл.

Следующие три буквы, которые идут после определения прав для владельца файла или каталога, означают права доступа для группы, которая владеет этим файлом.

В рассматриваемом нами примере выше у группы root будут права только на чтение файла.

Наконец, последние три буквы – это права доступа для всех остальных пользователей и групп, в том числе для абсолютно посторонних людей (если доступ в каталоги и файлы открыт на других ресурсах).

Изменение прав доступа.

Для назначения и изменения прав доступа используется команда chmod (сокращенно от change mode). Она вводится в командную строку по следующей логике:

chmod кто=права файл/каталог

Вместо «кто» вам нужно подставить обозначение того, для кого будет назначены права доступа. Существуют следующие обозначения:

- u – user – владелец файла;
- g – group – группа, которой принадлежит файл;
- o – other – остальные пользователи;
- a – all – все (вместо сочетания ugo).

Далее вместо «права» вам нужно ввести обозначение права, которое вы хотите дать этому пользователю или группе. О существующих правах уже было рассказано выше. Эти права будут даны вместо имеющихся.

Наконец, вместо «файл/каталог» вам нужно написать в команде название файла или каталога, права к которому вы хотите изменить.

Допустим, вы хотите, чтобы файл logs все сторонние (остальные) пользователи могли только читать. Для этого вам нужно ввести вот такую команду:

```
chmod o=r logs
```

Даже если до этого у пользователей было больше прав (например, они могли еще и изменять файл), то теперь у них останется только право на чтение:

```
-rwxr-xr—-
```

Вы можете комбинировать обозначения тех, кому хотите изменить права доступа. Например, если нужно изменить права доступа сразу и для владельца, и для группы, то можно написать вот так:

```
chmod ug=rw file1.txt
```

В этом случае владелец и группа смогут читать и изменять файл file1.txt.

Комбинировать можно и сами файлы или каталоги (если вы хотите изменить права сразу для нескольких файлов):

```
chmod ug=rw file1.txt file2.txt
```

Добавление и исключение прав

Еще один интересный нюанс – вы можете также использовать знаки плюса (+) и минуса (-). Это полезно в тех случаях, когда вы хотите предоставить (добавить) или убрать (лишить) какие-либо права.

К примеру, команда

```
chmod o+w file.txt
```

даст остальным пользователям возможность редактировать этот файл.

Копирование прав.

Также можно копировать (передавать) права доступа между разными пользователями. Допустим, вам нужно, чтобы остальные пользователи имели такие же права, как и владелец файла. Тогда вам нужно ввести следующую команду:

```
chmod o=u file.txt
```

Но менять сами права при вводе команды такой конструкции нельзя.

Цифровое обозначение.

Наверняка вы ни раз сталкивались с тем, что папкам или файлам даются права доступа в виде цифр. Например, 754, 755, 774 и т.д.

Каждая из цифр – это то же обозначение прав доступа для владельца, группы и остальных пользователей соответственно.

Расшифровка: чтение (r) – 4, запись (w) – 2 и выполнение (x) – 1. Если сложить все эти права, то получится 7 – такое право доступа может быть у владельца файла. Группа может иметь право на чтение и запись (4+2) – обозначается 1. И так далее.

Чтобы было понятнее:

- 7 – r+w+x – чтение, запись, выполнение;
- 6 – r+w – чтение и запись;
- 5 – r+x – чтение и выполнение;
- 4 – r – чтение;
- 3 – w+x – запись и выполнение;
- 2 – w – запись;
- 1 – x – выполнение;
- 0 – отсутствие каких-либо прав.

Такая запись пошла из двоичного кодирования восьмеричных цифр, то есть 754 – это восьмеричная запись 9 бит, которые задают права для файла или каталога.

При желании вы можете использовать команду chmod с цифровым кодированием:

```
chmod 754 file1.txt
```

Примеры:

644 – владелец файла может читать и изменять файл, а остальные пользователи (в том числе и группа) – только читать;

777 – все пользователи могут читать, изменять и выполнять файл.

Помните, что права доступа всегда выставляются от владельца файла к группе файла, а затем к остальным пользователя; то есть больше всего прав (или хотя бы точно не меньше) должно быть у владельца.

Настройка пользователей и прав доступа в MySQL.

Создание нового пользователя происходит в среде mysql с помощью команды CREATE USER:

```
CREATE USER 'username'@'host' IDENTIFIED BY 'password';
```

Где:

1. 'username' — имя создаваемого пользователя.
2. 'host' — хост, с которого пользователь может подключаться. Укажите localhost для локальных подключений или % для подключений с любых хостов. Также вы можете установить любой IP-адрес или доменное имя.
3. 'password' — пароль для нового пользователя.

Например, для создания пользователя user с возможностью локального подключения и заданным паролем password необходимо ввести:

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'password';
```

Для сравнения создадим пользователя user1 с тем же паролем и возможностью подключения с любого хоста (рис. 41):

```
CREATE USER 'user1'@'%' IDENTIFIED BY 'password';
```

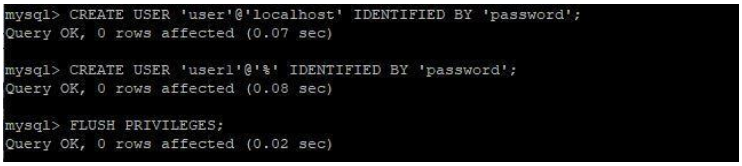


Рис. 41

Чтобы убедиться, что пользователь был успешно создан, вы можете выполнить запрос к системной таблице mysql.user (рис. 42):

```
SELECT User, Host FROM mysql.user;
```

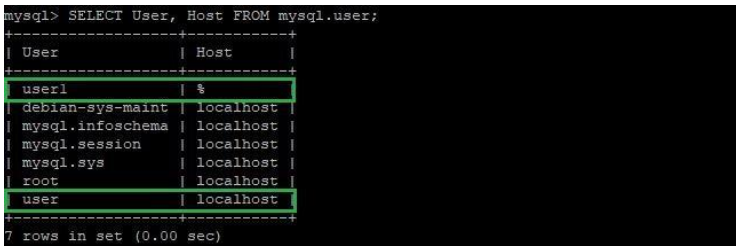


Рис. 42

Добавление прав.

Когда пользователь создан, можно переходить к назначению прав (привилегий). Все привилегии устанавливаются с помощью команды GRANT на уровне сервера. Также эта команда используется для добавления дополнительных привилегий к уже существующим для определенного пользователя.

Синтаксис запроса выглядит следующим образом:

```
GRANT privileges ON database.table TO 'username'@'host';
```

Если требуется предоставить привилегии для всех баз данных или таблиц, замените их имена звездочкой — «*». Чтобы установить привилегии на уровне столбцов, необходимо в скобках указать названия столбцов рядом с каждой соответствующей привилегией:

```
GRANT privilege (column1, column_2 ...) ON database.table TO 'username'@'host';
```

В MySQL привилегии можно назначать для всех баз данных и объектов, а также для конкретных баз данных, таблиц, столбцов или процедур. Это позволяет гибко управлять доступом пользователей. Рассмотрим наиболее частые команды:

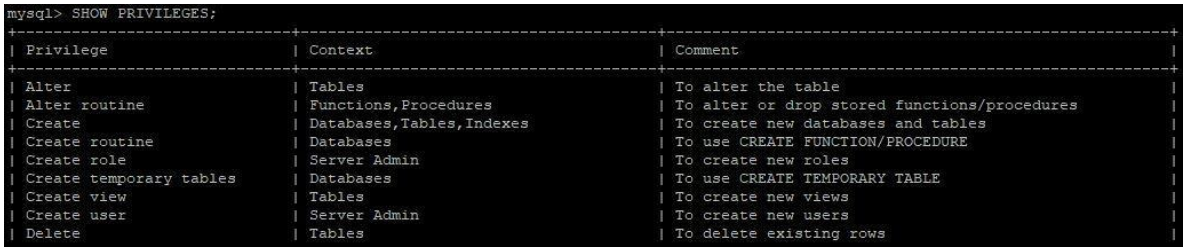
- Создание и удаление баз данных и таблиц — CREATE и DROP.
- Изменение структуры базы данных (добавление, удаление или изменение столбцов и таблиц) — ALTER.
- Предоставление привилегий другим пользователям — GRANT OPTION.

- Предоставление права выполнения административных команд — SUPER.
- Чтение из, запись в и изменение таблиц в данной базе данных — SELECT, INSERT, UPDATE, DELETE.
- Создание и удаление индексов в таблицах данной базы данных — INDEX.
- Вызов хранимых процедур и функций в данной базе данных — EXECUTE.

Отметим, что в качестве примера рассмотрены лишь некоторые из множества привилегий. С полным списком доступных прав можно ознакомиться при помощи команды:

```
SHOW PRIVILEGES;
```

В ответ инструмент отобразит все возможные привилегии и контекст их применения (рис. 43):



```
mysql> SHOW PRIVILEGES;
+-----------------------------+-----------------------------------------+-----------------------------------------------------------+
| Privilege                   | Context                                 | Comment                                                   |
+-----------------------------+-----------------------------------------+-----------------------------------------------------------+
| Alter                       | Tables                                  | To alter the table                                        |
| Alter routine               | Functions,Procedures                    | To alter or drop stored functions/procedures              |
| Create                      | Databases,Tables,Indexes                | To create new databases and tables                        |
| Create routine              | Databases                               | To use CREATE FUNCTION/PROCEDURE                          |
| Create role                 | Server Admin                            | To create new roles                                       |
| Create temporary tables     | Databases                               | To use CREATE TEMPORARY TABLE                             |
| Create view                 | Tables                                  | To create new views                                       |
| Create user                 | Server Admin                            | To create new users                                       |
| Delete                      | Tables                                  | To delete existing rows                                   |
```

Рис. 43

Давайте рассмотрим пример предоставления наиболее используемых привилегий. За основу возьмем пользователя user2:

```
CREATE USER 'user2'@'localhost' IDENTIFIED BY 'password';
```

Создадим базу данных user2db:

```
CREATE DATABASE user2db;
```

Создадим таблицу user2table:

```
USE user2db;
```

```
CREATE TABLE user2table (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255));
```

Осталось выдать права для работы с новой таблицей. Установим привилегии SELECT, INSERT, UPDATE, DELETE:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON user2db.user2table TO 'user2'@'localhost';
```

Таким образом пользователь получает права на чтение, вставку, обновление и удаление данных в определенной таблице базы данных.

Отзыв привилегий.

Процесс удаления разрешений происходит с использованием команды REVOKE. Она позволяет удалить одно или несколько разрешений для конкретного пользователя или роли. Синтаксис команды:

```
REVOKE privilege ON database.table FROM 'username'@'host';
```

Для отзыва нескольких привилегий сразу укажите их через запятую:

```
REVOKE privilege 1, privilege 2, ..., Privilege N ON database.table FROM 'username'@'host';
```

Для отзыва всех привилегий для конкретного пользователя запрос имеет вид:

```
REVOKE ALL PRIVILEGES ON database.table FROM 'username'@'host';
```

Например, чтобы отозвать привилегии, которые мы давали новому пользователю ранее, нужно использовать следующую команду:

```
REVOKE ALL PRIVILEGES ON user2db.user2table FROM 'user2'@'localhost';
```

Выполним данную команду и посмотрим на результат:

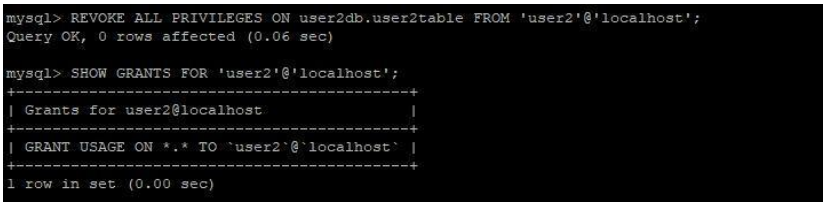


Рис. 44

Просмотр привилегий пользователей.

Для просмотра прав используется команда:

```
SHOW GRANTS FOR 'username'@'host';
```

Где 'username'@'host' — это пользователь и его хост, для которого вы хотите увидеть привилегии.

Выполним команду для проверки прав нашего пользователя:

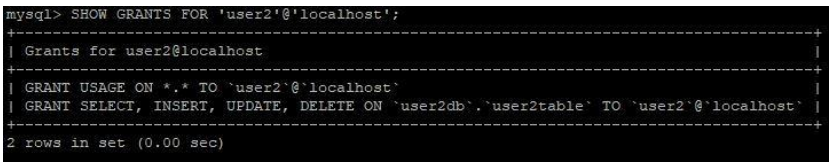


Рис. 45

Первая строка указывает, что у пользователя 'user2'@'localhost' есть право USAGE на все базы данных и таблицы. Это право является стандартным для новых пользователей и предоставляется без дополнительного указания. Вторая строка указывает на привилегии, которые мы установили ранее.

Создание суперпользователя.

Суперпользователь в MySQL обладает полными привилегиями и имеет доступ ко всем базам данных, что может повлечь за собой риски безопасности. Рекомендуется создавать суперпользователя только при необходимости.

Ранее мы создали простого пользователя с именем «user», сделаем из него суперпользователя. Для этого воспользуемся командой:

```
GRANT ALL PRIVILEGES ON *.* TO 'user'@'localhost' WITH GRANT OPTION;
```

После этого проверим привилегии суперпользователя при помощи запроса:

```
SHOW GRANTS FOR 'user'@'localhost';
```

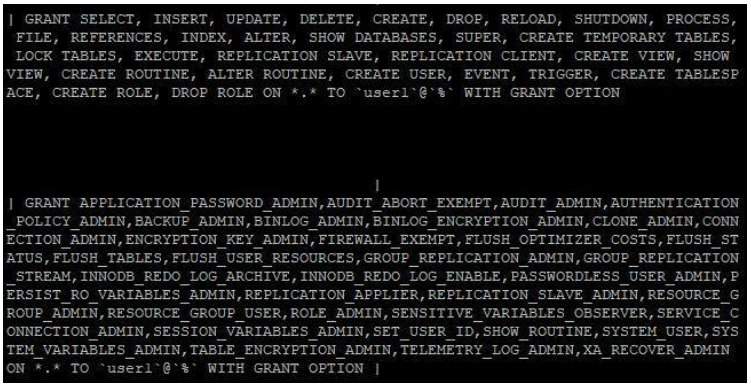


Рис. 46

Для сравнения посмотрим на привилегии простого пользователя:

```
SHOW GRANTS FOR 'user1'@'%';
```

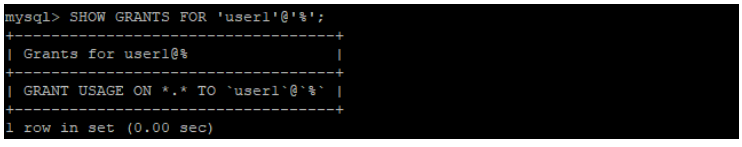


Рис. 47

В данном случае привилегии полностью отсутствуют, имеются только права на подключение к серверу MySQL.

Переименование пользователя.

Изменение имени пользователя в MySQL представляет собой простой и эффективный способ обновить идентификатор пользователя без потери его прав доступа. Это полезно, когда необходимо изменить имя хоста или имя пользователя для входа в базу данных. Благодаря этой операции вы избегаете необходимости создавать нового пользователя и вручную настраивать его привилегии.

Для выполнения переименования пользователя в MySQL используется RENAME USER. У команды существует 2 ограничения:

1. Необходимо иметь глобальное право CREATE USER или право UPDATE на системную схему Mysql. Если переменная системы read_only включена, команда RENAME USER также требует права CONNECTION_ADMIN (или устаревшего права SUPER).
2. Новое имя пользователя не должно уже существовать в базе данных.

Синтаксис команды выглядит следующим образом:

```
RENAME USER old_username TO new_username;
```

Для примера переименуем пользователя user в newuser. Уберем права суперпользователя, которые дали ранее:

```
REVOKE ALL PRIVILEGES ON *.* FROM 'user'@'localhost';
```

Произведем переименование:

```
RENAME USER 'user'@'localhost' TO 'newuser'@'localhost';
```

Проверим результат при помощи команды:

```
SELECT User, Host FROM mysql.user;
```

```
mysql> SELECT User, Host FROM mysql.user;
+------------------+-----------+
| User             | Host      |
+------------------+-----------+
| user1            | %         |
| debian-sys-maint | localhost |
| mysql.infoschema | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| newuser          | localhost |
| root             | localhost |
| user2            | localhost |
+------------------+-----------+
8 rows in set (0.01 sec)
```

Рис. 48

Также можно производить операцию по переименованию, когда необходимо заменить хост, с которого пользователь может подключаться. Для этого оставьте имя пользователя прежним, но замените хост в запросе. Например:

```
RENAME USER 'newuser'@'localhost' TO 'newuser'@'%';
```

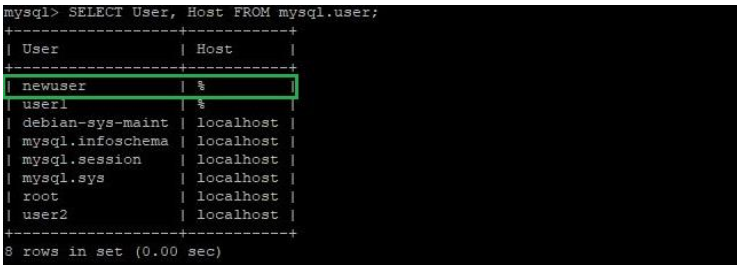


Рис. 49

Удаление пользователя MySQL.

Доступно при помощи команды DROP USER. Будьте внимательны при выполнении этой операции. Она выполняется без дополнительного подтверждения и способна удалить как простого пользователя, так и суперпользователя.

Синтаксис выглядит так:

```
DROP USER 'username'@'host';
```

Удалим пользователя newuser:

```
DROP USER 'newuser'@'%';
```

Проверим список пользователей после операции:

```
SELECT User, Host FROM mysql.user;
```

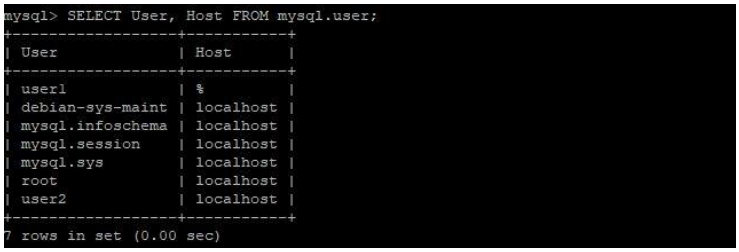


Рис. 50

Последовательность выполнения лабораторной работы

1. Настройте права доступа к файлам вашего сайта.

2. Настройте права доступа к пользователям вашей БД MySQL.

3. Показать результаты работы преподавателю.

4. Ответить на вопросы преподавателя.

Лабораторная работа №14 «Настройка ролей доступа пользователей в CMS»

Теоретические сведения

Рассмотрим настройку ролей доступа на примере популярной CMS Wordpress.

WordPress использует механизм Ролей пользователей, суть которого заключается в том, чтобы дать возможность контроля, что другие пользователи сайта могут или не могут делать в админке, например публиковать свои посты, создавать страницы, устанавливать плагины, темы, редактировать других пользователей и так далее.

У каждой роли есть свой набор прав, о которых мы поговорим чуть ниже, к примеру «Администраторы» — это группа пользователей, а switch_themes (возможность смены темы оформления) уже относится к правам этой группы.

В WordPress по умолчанию уже существует 6 групп (ролей) пользователей:

- Super Admin — суперадминистратор, который имеет право управлять сетью сайтов.
- Administrator — администратор.
- Editor — редактор, может публиковать и редактировать посты других пользователей.
- Author — автор, может публиковать и редактировать собственные посты.
- Contributor — участник, может писать и отправлять свои посты на модерацию.
- Subscriber — подписчик, всё, что он может — это редактировать свой профиль.

Сразу после установки WordPress автоматически создается пользователь-администратор.

Также вы можете установить, какую роль нужно присваивать только что зарегистрированному пользователю. Это настраивается в «Настройки > Общие» (рис. 51).

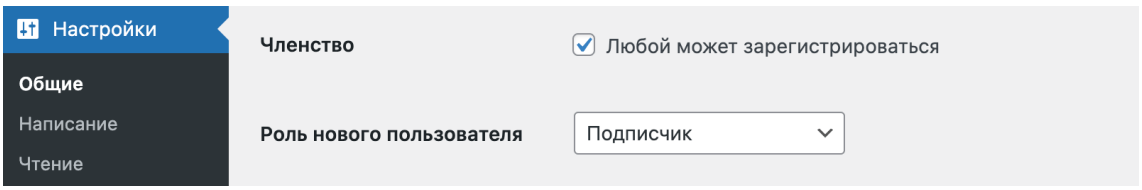


Рис. 51. Права доступа в Wordpress

Изменить роль пользователя можно на странице его профиля или же на странице со всеми пользователями (рис. 52).

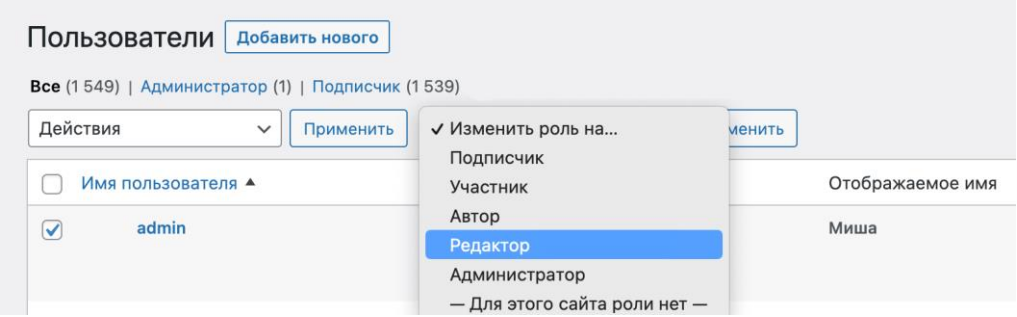


Рис. 52. Изменения роли доступа

Администратор (administrator).

Вот так выглядит админка администраторов (рекомендую обратить внимание именно на меню слева) (рис. 53).

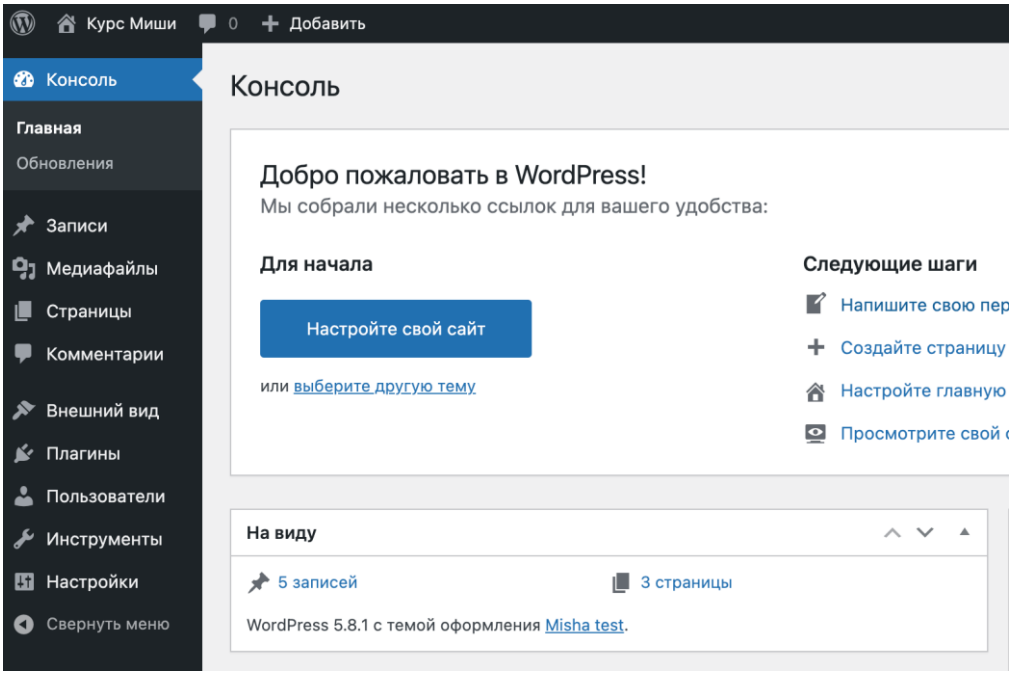


Рис. 53. Панель администратора

Если мы не говорим о сети сайтов WordPress Мультисайт, то администраторы могут всё – устанавливать плагины и темы, редактировать весь контент и настройки, создавать и удалять пользователей, сбрасывать пароли и так далее.

Редактор (editor).

Редакторы – это такие ребята, которые не имеют доступ к настройкам и плагинам сайта, зато имеют полный доступ к любому контенту и могут создавать и редакторивать как свои посты, так и чужие. А также модерировать комментарии и редактировать категории и метки (рис. 54).

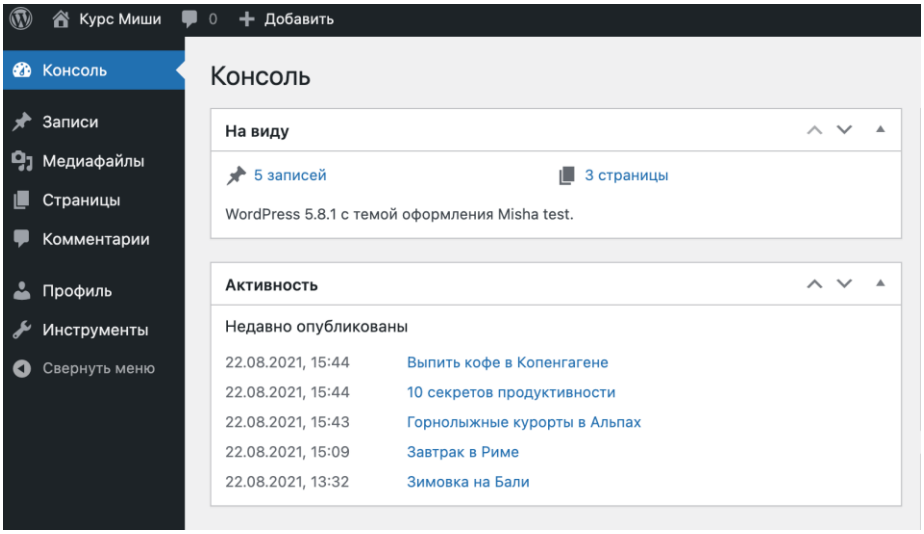


Рис. 54. Консоль редактора

Автор (author).

Авторы могут публиковать и редактировать только свои собственные посты типа «Записи». И удалять кстати тоже. То есть у них уже нет доступа к Страницам. И не могут создавать новые рубрики и метки (рис. 55).

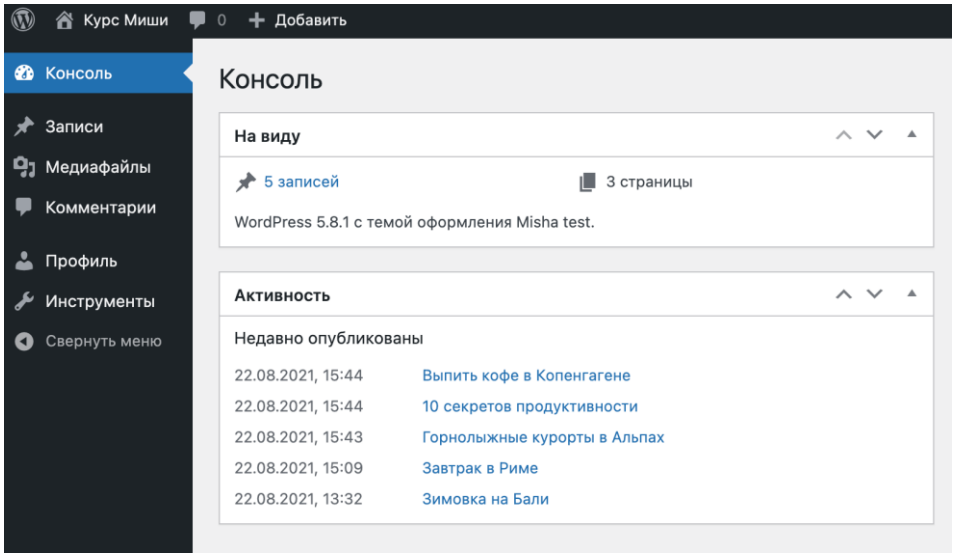


Рис. 55. Консоль автора

Участник (contributor).

Что-то типо облегчённой версии роли автора. Могут писать и редактировать посты в виде черновиков, но при публикации посты должны

пройти одобрение администраторами или редакторами. Кроме того, не могут изменить свои же, уже опубликованные посты и загружать медиафайлы (рис. 56).

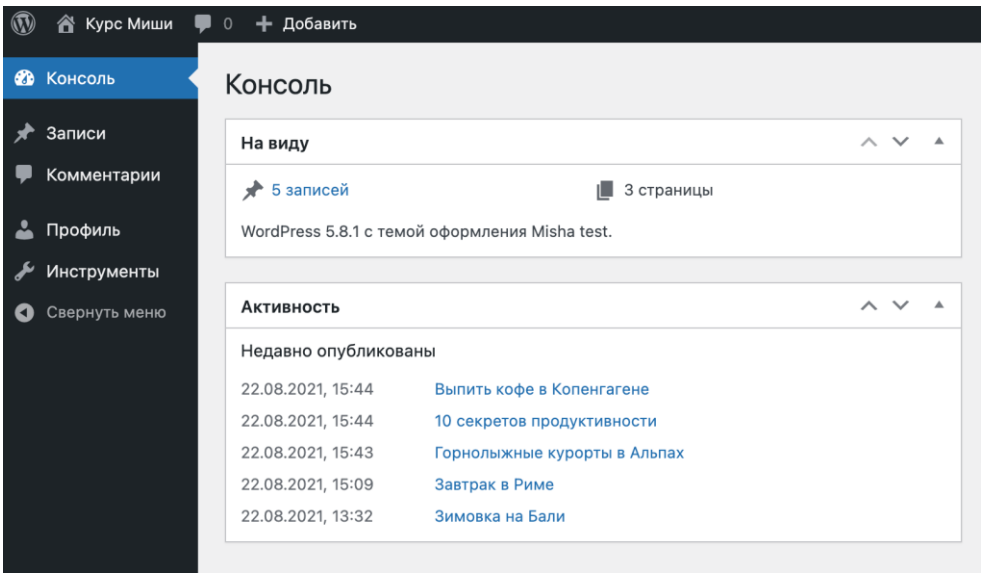


Рис. 56. Консоль участника

Подписчик (subscriber).

Ну и последняя стандартная роль WordPress – подписчик, который может только изменять информацию в своём профиле и читать посты, на этом всё (рис. 57).

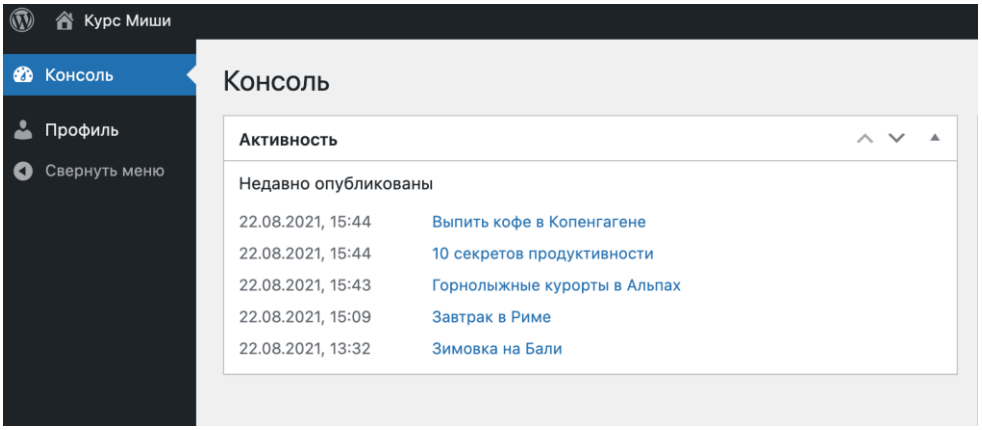


Рис. 57. Консоль подписчика

Супер-администратор.

Вы наверное подумали: «Так, а где роль суперадмина?» Почему я не рассказал про неё в самом начале? А это даже и не совсем роль – это роль администратора с возможностью управления сетью WordPress Мультисайт.

У него появляется доступ в отдельную консоль для мультисайта (рис. 58).

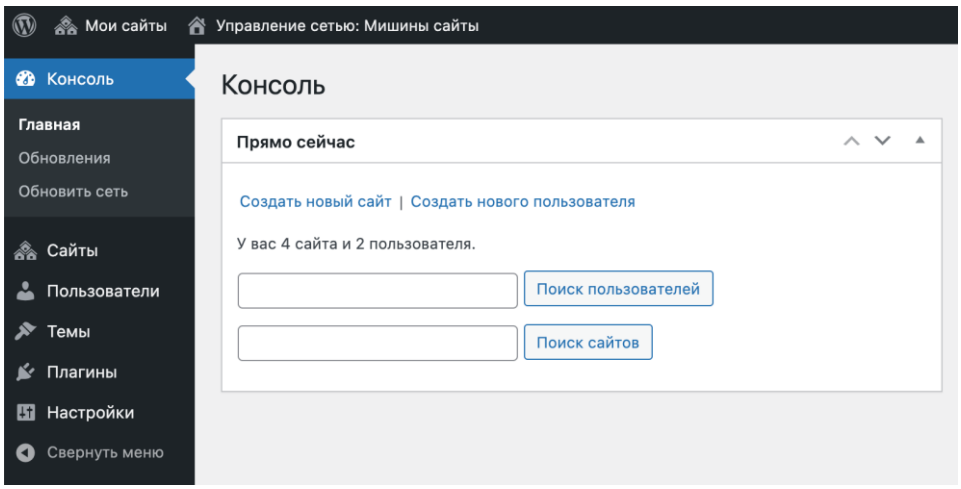


Рис. 58. Консоль суперадминистратора

Примитивные и Мета-права.

Сейчас пришло время рассказать вам о важном моменте, связанном с правами пользователя. Мы также упоминали о нём, когда описывали функцию register_post_type().

Чтобы понять, как это работает, лучше всего обратиться к примеру.

Предположим, что у нас есть какое-то право, допустим это edit_posts. И мы например можем проверить функцией current_user_can(), может ли текущий пользователь в принципе редактировать посты или нет. Это примитивное право, под него попадают все возможные посты. Это можно проверить:

```
if( current_user_can( 'edit_posts' ) ) {


}
```

Но как вы знаете, у нас есть например роль Авторы, которые могут редактировать только свои посты. Как проверить, что они это могут делать? При помощи мета-права!

```
if( current_user_can( 'edit_post', $post->ID ) ) {


}
```

Мета-права создаются «на лету» динамически, например алгоритм такой, что WordPress чекает, может ли пользователь редактировать чужие посты примитивным правом `edit_others_posts`, если нет, то он проверяет, является ли пользователь автором проверяемого поста `$post->ID`. И вот так это и работает.

Проверка роли или права пользователя.

Для того, чтобы в коде проверить, имеет ли пользователь сайта определённую роль или же только определённое право, мы можем воспользоваться одной из двух функций:

- user_can() – проверяет пользователя, ID которого передан в параметрах.
- current_user_can() – проверяет текущего пользователя.

Проверим, что текущий пользователь администратор:

```
if( current_user_can( 'administrator' ) ) {
        // выполняем какие-либо действия
}
```

Или:

```
if( user_can( get_current_user_id(), 'administrator' ) ) {
        // выполняем какие-либо действия
}
```

Таким же образом вместо названия роли пользователя мы можем передать и название определённого права. Например проверим, что пользователь может переключать темы:

```
if( current_user_can( 'switch_themes' ) ) {
        // выполняем какие-либо действия
}
```

И ещё кое-что, так как супер-администратор не является полноценной ролью, то его лучше проверять по-другому, функцией is_super_admin() или правом setup_network.

```
if( is_super_admin() ) { // или if( current_user_can( 'setup_network' ) ) {
        // в параметры также можем передать ID определённого пользователя
}
```

add_role() — создание собственной роли. Функция заносит данные в базу, поэтому лучше всего её использовать только один раз, например при активации плагина или темы.

```
    /*
     * допустим я добавлю этот код в файл плагина и сделаю так, чтобы он запускался при активации этого самого плагина
     */
    register_activation_hook( __FILE__, 'true_new_role_plugin_activate' );

    function true_new_role_plugin_activate() {
            $new_role = add_role(
                    'comm_moderator', // название роли
                    __( 'Comment Moderator' ), // отображаемое название роли (модератор комментариев)
                    array( // массив возможностей, true - разрешено, false - запрещено
                            'read'          => true,  // ну это понятно
                            'moderate_comments'=> true // разрешим модерировать комментарии
                    )
            );
            if ( null !== $result ) { // смотрим результат
                    // роль успешно создана
            } else {
                    // если null, то значит роль уже существует
            }
    }
```

remove_role() — удаление ролей. Также, как и `add_role()`, функция изменяет содержимое базы данных — а значит не нужно просто тупо вставлять её в `functions.php`.

В примере удалим роль, созданную в прошлой главе:

```
remove_role('comm_moderator');    //    в    качестве
параметра указываем название роли и всё, дело сделано
```

get_role() — получение информации о группе пользователей. В случае успеха возвращает объект WP_Role (который состоит преимущественно из возможностей роли), в случае неудачи — null.

```
$my_role = get_role( 'comm_moderator' ); //
указываем роль, которая нам нужна

print_r( $my_role ); // так можно вывести
содержимое объекта
```

add_cap() и remove_cap() — добавление и удаление прав. Благодаря этим функциям вы можете добавить или удалить права для пользователей определенной роли или даже для пользователей с определенными ID.

Эти функции также изменяют содержимое базы данных, поэтому в качестве примера мы повесим их на активацию / деактивацию темы.

```
function true_author_caps(){

        global $pagenow;

        $role = get_role( 'author' ); // к примеру
возьмем роль автора

        // $role = new WP_User( $user_id ); таким
образом мы можем взять конкретного пользователя

        if ( 'themes.php' == $pagenow && isset(
$_GET['activated'] ) ){ // если тема была активирована

                $role->add_cap( 'edit_others_posts'
); // разрешаем авторам редактировать посты других
авторов

        } else { // если тема деактивирована

                $role->remove_cap(
'edit_others_posts' );

        }

}


add_action( 'load-themes.php', 'true_author_caps'
); // вешаем функцию на хук
```

Последовательность выполнения лабораторной работы

1. Установите CMS Wordpress.

2. Создайте пользователей типа Редактор, Автор, Участник, Подписчик.

3. Проверьте разрешенные действия для каждого типа пользователей.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №15 «Анализ безопасности веб-сервиса на предмет наличия уязвимостей»

Теоретические сведения

Проверка безопасности веб-сайта – неотъемлемый процесс разработки и поддержки веб-проектов. Важно понимать, какие уязвимости могут быть доступны злоумышленникам и как их предотвратить.

Основные инструменты и методы проведения проверки безопасности веб-сайта:

- Сканирование уязвимостей: специальное программное обеспечение позволяет автоматически обнаруживать слабые места в защите веб-сайта;
- Анализ кода: проверка безопасности кода веб-сайта на наличие уязвимостей, таких как SQL-инъекции или XSS-атаки;
- Пентестинг: проверка безопасности путем моделирования атаки на веб-сайт, чтобы выявить его уязвимости и способы их эксплуатации;
- Мониторинг безопасности: непрерывный контроль за активностью и уязвимостями веб-сайта, а также системы предупреждения об атаках.

Проведение систематической проверки безопасности веб-сайта – это обязательный этап для защиты вашего онлайн-проекта. Не забывайте о необходимости обновления программного обеспечения, установки надежных паролей и резервного копирования данных, чтобы обеспечить безопасность вашего веб-сайта.

Основные инструменты и методы проверки безопасности веб-сайта.

Рассмотрим основные инструменты и методы проверки безопасности вашего веб-сайта. Следуя этим рекомендациям, вы сможете повысить надежность своего сайта и защитить его от потенциальных угроз.

1. Анализ уязвимостей.

Первым шагом в проверке безопасности веб-сайта является анализ его уязвимостей. Существует множество инструментов и сервисов, которые помогают обнаружить возможные проблемы, такие как устаревшие версии ПО, незащищенные точки входа или слабые пароли.

Некоторые из популярных инструментов для анализа уязвимостей веб-сайта включают Nessus, OpenVAS и Nikto. Они обеспечивают сканирование сайта на предмет известных уязвимостей и предоставляют подробные отчеты о найденных проблемах.

2. Тестирование на проникновение.

Тестирование на проникновение (penetration testing) является следующим важным шагом в проверке безопасности веб-сайта. Оно позволяет осуществить моделирование реальных атак на сайт с целью проверки его устойчивости к подобным атакам.

Профессиональные тестировщики на проникновение используют специальные инструменты и методы, чтобы определить уязвимости и потенциальные точки входа, которые могут быть использованы злоумышленниками для получения несанкционированного доступа.

3. Обновление ПО и патчи безопасности.

Регулярное обновление ПО и установка патчей безопасности - это один из самых простых и важных способов обеспечить безопасность вашего веб-сайта.

Постоянно следите за выходом обновлений от разработчиков используемого ПО и операционной системы. Установка свежих версий и патчей заполняет известные уязвимости и защищает ваш сайт от известных атак.

4. Использование сильных паролей.

Использование сложных и уникальных паролей для всех учетных записей на вашем сайте - важный момент. Простые пароли и повторное использование паролей ставят ваш сайт под угрозу взлома.

Рекомендуется использовать длинные пароли, состоящие из комбинации букв (в верхнем и нижнем регистре), цифр и специальных символов. Также важно регулярно менять пароли пользователям и администраторам сайта.

5. Защита от SQL-инъекций и скриптовых атак.

SQL-инъекции и скриптовые атаки являются распространенными видами атак на веб-сайты. Уязвимые точки входа могут позволить злоумышленникам выполнить вредоносные операции на вашем сервере или получить доступ к вашей базе данных.

Чтобы защитить свой сайт от таких атак, рекомендуется использовать параметризованные запросы и фильтрацию вводимых данных. Также необходимо правильно настраивать ваш сервер и использовать специальные инструменты для обнаружения и блокирования подобных атак.

Безопасность веб-сайта - это постоянный процесс, и эти основные инструменты и методы помогут вам поддерживать безопасность вашего сайта на высоком уровне.

Перечень инструментов.

1. Сканеры безопасности.

Сканеры безопасности являются одним из основных инструментов при проведении проверки безопасности веб-сайта. Они позволяют автоматизировать процесс обнаружения уязвимостей и проверки соответствия сайта стандартам безопасности. Некоторые популярные сканеры безопасности включают в себя:

- OWASP ZAP - это свободно распространяемый инструмент с открытым исходным кодом, предназначенный для проведения тестирования на проникновение и проверки безопасности веб-приложений;
- Nessus - это коммерческий сканер безопасности, известный своей высокой производительностью и способностью обнаруживать различные уязвимости в системе;
- Nikto - это бесплатный сканер безопасности, специализирующийся на обнаружении уязвимостей в веб-серверах и веб-приложениях.

2. Анализаторы кода.

Анализаторы кода помогают обнаружить уязвимости в исходном коде веб-приложений. Они анализируют код на предмет наличия уязвимых мест, включая потенциальные проблемы с безопасностью. Некоторые популярные анализаторы кода включают в себя:

- Fortify - это коммерческий инструмент, предназначенный для статического анализа исходного кода приложений на предмет наличия уязвимостей.
- SonarQube - это бесплатный инструмент с открытым исходным кодом, который предоставляет возможность анализировать код и выявлять наличие уязвимостей в нем.
- Veracode - это коммерческий инструмент, специализирующийся на поиске уязвимостей в исходном коде приложений любого языка программирования.

3. Web Application Firewall (WAF).

Web Application Firewall (WAF) - это межсетевой экран, специально разработанный для защиты веб-приложений от различных атак. Он обеспечивает высокий уровень безопасности, блокируя попытки эксплойтов и вредоносных атак. Некоторые известные WAF включают в себя:

- ModSecurity - это бесплатный исходный код, который может быть установлен на сервере и предоставляет возможность встраивать защиту от веб-уязвимостей веб-приложений.

- Barracuda Web Application Firewall - это коммерческое решение, предоставляющее высокий уровень защиты от различных типов атак, включая SQL-инъекции и кросс-сайтовый скриптинг (XSS).

- Cloudflare WAF - это облачный сервис, который обеспечивает защиту от атак и уязвимостей для веб-приложений без необходимости установки дополнительного оборудования.

Портсканер.

Портсканеры могут использоваться как для легальных целей, например, анализа безопасности собственного сервера или сети, так и в злонамеренных целях, в том числе для поиска уязвимостей и взлома систем. Поэтому важно использовать портсканеры только с разрешения владельца сети или системы, чтобы избежать нарушения закона.

Существует множество портсканеров, но самые распространенные – это Nmap и Nessus. Nmap является открытым программным обеспечением и предоставляет большое количество функций для сканирования портов. Nessus – это платная программа, которая помимо сканирования портов предлагает широкий спектр возможностей для обнаружения уязвимостей и выполнения тестов на проникновение.

Применение портсканера позволяет обнаружить открытые порты и оценить уровень безопасности системы. Например, если на сервере обнаружены открытые порты, на которых работают сервисы, необходимо проверить, имеются ли у них уязвимости и принимаются ли меры по их обеспечению. Также, портсканер может быть полезен при настройке новой сети или сервера, помогая выявить проблемы и необходимые настройки.

- Nmap - Открытое программное обеспечение, предоставляющее функции для сканирования портов;

- Nessus - Платная программа с возможностями обнаружения уязвимостей и тестирования на проникновение.

Детектор уязвимостей.

Одним из главных преимуществ использования детектора уязвимостей является его автоматизированный характер. Программа сама производит сканирование веб-сайта на предмет уязвимостей, что значительно упрощает процесс и экономит время. Для работы детектора уязвимостей не требуется специальных знаний в области безопасности – достаточно лишь указать адрес веб-сайта, который нужно проверить.

В зависимости от выбранного детектора уязвимостей, он может обнаруживать различные виды уязвимостей, такие как уязвимости XSS (межсайтовый скриптинг), SQL-инъекции, переполнения буфера и другие. Данная информация помогает разработчикам и администраторам принять необходимые меры по устранению обнаруженных уязвимостей и повысить уровень безопасности веб-сайта.

После завершения сканирования, детектор уязвимостей предоставляет детальный отчет о найденных уязвимостях. В отчете указывается тип уязвимости, место ее обнаружения, а также предлагаются рекомендации по устранению уязвимости. Это позволяет оперативно принимать меры по защите веб-сайта и предотвращать возможные атаки.

Использование детектора уязвимостей является важной составляющей в обеспечении безопасности веб-сайта. Регулярное сканирование веб-сайта на предмет уязвимостей помогает предотвратить возможные атаки и сохранить данные пользователей и веб-сайта в целости и сохранности.

Инструменты анализа кода.

Анализ кода веб-сайта позволяет выявить потенциальные уязвимости и проблемы безопасности. Существуют различные инструменты, которые помогают провести такой анализ:

- Сканеры уязвимостей - программы, которые автоматически сканируют код и ищут уязвимости. Они могут обнаружить такие проблемы, как отсутствие фильтрации ввода, небезопасное хранение данных или уязвимости связанные с авторизацией и аутентификацией.

- Статические анализаторы кода - инструменты, которые анализируют исходный код без его выполнения. Они ищут потенциальные ошибки и проблемы безопасности, такие как неправильное использование криптографических функций, небезопасные операции с памятью или несанкционированный доступ к данным.

- Динамические анализаторы кода - инструменты, которые анализируют код во время его выполнения. Они могут обнаружить уязвимости, которые проявляются только при определенных условиях или взаимодействии с другими компонентами системы.

Выбор конкретных инструментов зависит от требований проекта и доступных ресурсов. Рекомендуется использовать несколько инструментов для максимального охвата потенциальных проблем безопасности.

После проведения анализа кода следует исправить обнаруженные уязвимости и проблемы безопасности. Это может включать изменение кода, обновление используемых библиотек или настройку системы.

Важно помнить, что анализ кода - это не конечный процесс, и он должен проводиться регулярно. Такие инструменты, как сканеры уязвимостей и анализаторы кода, помогают сделать веб-сайт более безопасным и защитить его от потенциальных атак.

Методы проверки.

Метод проверки 1: Сканирование уязвимостей.

Одним из основных методов проверки безопасности веб-сайта является сканирование уязвимостей. Это процесс, в ходе которого специальные инструменты проверяют наличие возможных уязвимостей, которые могут быть использованы злоумышленниками для атаки на ваш сайт. Сканирование уязвимостей позволяет обнаружить такие проблемы и принять меры для их устранения.

Метод проверки 2: Тестирование на проникновение.

Тестирование на проникновение (pentest) – это процесс, в ходе которого проверяются различные аспекты безопасности веб-сайта путем попыток несанкционированного проникновения. Специалисты по безопасности используют различные методы и инструменты, чтобы выявить потенциальные уязвимости и оценить степень защиты сайта от атак.

Метод проверки 3: Анализ кода.

Анализ кода является важной частью проверки безопасности веб-сайта. При анализе кода специалисты проанализируют все компоненты сайта, включая главную страницу, скрипты, базу данных и т.д. Целью анализа кода является выявление потенциальных уязвимостей, связанных с программным кодом, и предложение рекомендаций по устранению этих проблем.

Метод проверки 4: Социальная инженерия.

Социальная инженерия – это метод проверки безопасности, который заключается в исследовании и тестировании человеческого фактора. Злоумышленники могут попытаться получить доступ к вашему веб-сайту, обманув сотрудников или пользователей. В ходе проверки специалисты анализируют различные сценарии и создают эксперименты, чтобы определить, насколько уязвимы сотрудники и пользователи вашего сайта перед социальными атаками.

Метод проверки 5: Мониторинг безопасности.

Мониторинг безопасности является важным методом проверки веб-сайта. Он позволяет отслеживать и реагировать на возможные атаки и уязвимости. Специалисты по безопасности могут использовать различные инструменты и сервисы для контроля активности веб-сайта, обнаружения подозрительных действий и оповещения о потенциальных угрозах.

Тестирование на проникновение.

Во время тестирования на проникновение эксперт по безопасности пытается проникнуть в систему, используя различные методы и инструменты. Он проводит подробный анализ системы, ищет уязвимости, провоцирует атаки и исследует реакцию системы на них. Это позволяет выявить уязвимые места и устранить их до того, как злоумышленники смогут нанести ущерб.

Основное внимание при тестировании на проникновение уделяется таким аспектам безопасности, как:

- Проверка на возможность несанкционированного доступа к системе и данным;
- Анализ уязвимостей и проверка их исправления;
- Проверка системы на предмет возможности атаки с целью перехвата данных;
- Проверка механизмов аутентификации и авторизации;
- Анализ защиты от различных видов атак, таких как XSS (межсайтовый скриптинг), SQL-инъекции, CSRF (межсайтовая подделка запроса) и др.;
- Анализ шифрования данных и проверка на уязвимости в криптографии.

В процессе тестирования на проникновение используются различные инструменты, включая сканеры уязвимостей, эксплойты, фаззеры и др. Эксперт по безопасности проводит тестирование, учитывая специфику системы, ее конфигурацию и особенности разработки.

Результатом тестирования на проникновение является отчет, в котором содержатся описания найденных уязвимостей, рекомендации по их устранению и общая оценка безопасности системы. По итогам исправления выявленных проблем система становится более устойчивой к атакам и предотвращает возможный ущерб.

Анализ логов сервера.

В процессе анализа логов сервера следует обратить внимание на следующие моменты:

1. Идентификация подозрительных IP-адресов. Если в логах обнаружена активность с неизвестных или подозрительных IP-адресов, это может указывать на попытку несанкционированного доступа к серверу.

2. Проверка HTTP-статусных кодов. Частые ошибки 404 (страница не найдена), 500 (внутренняя ошибка сервера) и другие могут говорить о наличии проблем на веб-сайте.

3. Анализ логов аутентификации. Если в логах обнаружена неудачная попытка аутентификации с определенного IP-адреса, это может свидетельствовать о попытке взлома учетной записи.

4. Исследование временных отметок. Анализ временных отметок в логах может помочь в обнаружении незапланированных или необычных активностей на сервере.

5. Анализ потенциально опасных URL-адресов. Проверка логов на наличие запросов к уязвимым или несуществующим страницам может помочь в обнаружении попыток эксплойтов.

Важно регулярно производить анализ логов сервера и реагировать на обнаруженные угрозы безопасности веб-сайта. Современные инструменты и методы анализа логов помогают повысить безопасность и защитить ваш сайт от внешних атак.

Последовательность выполнения лабораторной работы

1. Установите бесплатные инструменты анализа уязвимостей, например: OWASP ZAP, SonarQube, ModSecurity, Nmap.

2. Проанализируйте ваш сайт.

3. Опишите найденные уязвимости и методы их устранения.

4. Показать результаты работы преподавателю.

5. Ответить на вопросы преподавателя.

Лабораторная работа №16 «Настройка веб-сервера с использованием протокола HTTPS»

Теоретические сведения

Безопасный интернет давно перестал быть опцией — теперь это обязательное условие. Отсутствие HTTPS на вашем сайте не просто делает данные пользователей уязвимыми, но и отпугивает посетителей предупреждениями браузера о "небезопасном соединении". К тому же Google открыто использует HTTPS как фактор ранжирования. Возможно, именно поэтому вы здесь — чтобы наконец-то перевести свой сайт на защищенный протокол? Отлично! В этой инструкции я расскажу, как настроить HTTPS без лишней головной боли, даже если ваш опыт в администрировании сервера ограничен.

HTTPS (HyperText Transfer Protocol Secure) — это расширение стандартного HTTP-протокола с добавлением шифрования данных. Когда пользователь заходит на сайт с HTTPS, между его браузером и сервером устанавливается защищенное соединение, и вся передаваемая информация шифруется.

Основное отличие HTTPS от HTTP заключается в использовании криптографических протоколов SSL/TLS, которые обеспечивают три важных компонента безопасной коммуникации:

- Шифрование — никто не может прочитать ваши данные при передаче;
- Целостность данных — информация не может быть изменена или повреждена при передаче;
- Аутентификация — подтверждает, что ваши пользователи обмениваются информацией именно с вашим сайтом.

Почему HTTPS стал критически важным для любого сайта? Вот несколько ключевых причин (табл. 2).

С 2018 года Google Chrome маркирует все HTTP-сайты как "небезопасные". Firefox, Safari и другие браузеры придерживаются аналогичной политики. Это означает, что отсутствие HTTPS не просто ограничивает функциональность вашего сайта — оно активно отпугивает посетителей.

Выбор и получение SSL-сертификата для вашего сайта.

SSL-сертификат — это цифровой документ, подтверждающий подлинность вашего сайта и обеспечивающий шифрование данных. Выбор подходящего сертификата зависит от типа вашего сайта, бюджета и требуемого уровня защиты.

Существуют следующие основные типы SSL-сертификатов (табл. 3).

Таблица 2. Причины перехода на HTTPS

Причина	**Пояснение**	**Последствия игнорирования**
Безопасность данных	Шифрование всей информации между пользователем и сервером	Риск перехвата личных данных, паролей и платежной информации
Доверие пользователей	Индикатор "безопасно" в браузере и зеленый замок	Предупреждения о небезопасности и отток посетителей
SEO-преимущества	Google использует HTTPS как положительный фактор ранжирования	Снижение позиций в поисковой выдаче
Доступ к новым функциям	Некоторые современные веб-API доступны только через HTTPS	Ограничение в использовании передовых технологий

Таблица 3. Типы SSL-сертификатов

Тип сертификата	**Особенности**	**Уровень проверки**	**Идеально подходит для**	**Примерная стоимость/год**
DV (Domain Validation)	Базовое шифрование, быстрое получение	Проверка владения доменом	Блоги, информационные сайты	0-50$
OV (Organization Validation)	Проверка компании + шифрование	Проверка юридического лица	Бизнес-сайты, интернет-магазины	50-200$
EV (Extended Validation)	Максимальное доверие, зеленая адресная строка	Расширенная проверка компании	Финансовые сервисы, крупная электронная коммерция	200-500$
Wildcard	Защищает основной домен и все поддомены	Как DV или OV	Сайты с множеством поддоменов	100-500$
Multi-domain (SAN)	Один сертификат для нескольких доменов	Как DV, OV или EV	Компании с несколькими сайтами	100-1000$

Помимо платных вариантов, существует бесплатный SSL-провайдер Let's Encrypt, который предлагает DV-сертификаты с автоматическим обновлением каждые 90 дней. Это отличный вариант для большинства сайтов, не требующих подтверждения организации.

Процесс получения SSL-сертификата обычно включает несколько шагов:

1. Генерация CSR (Certificate Signing Request) на вашем сервере;
2. Отправка CSR в центр сертификации (CA);
3. Подтверждение владения доменом (через email, DNS-запись или файл);
4. Получение и установка сертификата на сервер.

Для генерации CSR на Linux-сервере используйте следующую команду:

```
openssl req -new -newkey rsa:2048 -nodes -keyout yourdomain.key -out yourdomain.csr
```

После выполнения команды вам потребуется ввести информацию о вашей организации. Полученный CSR-файл затем нужно предоставить центру сертификации. Если вы используете Let's Encrypt, процесс будет автоматизирован с помощью клиента Certbot. □

Установка SSL-сертификата на веб-сервер: Apache и Nginx.

После получения SSL-сертификата необходимо установить его на ваш веб-сервер. Процесс отличается в зависимости от того, какой сервер вы используете — Apache или Nginx.

Установка на Apache.

1. Разместите файлы сертификата на сервере. Обычно они включают:

- Сертификат домена (yourdomain.crt)
- Приватный ключ (yourdomain.key)
- Цепочка сертификатов (chain.crt), если предоставлена

2. Активируйте модуль SSL для Apache:

```
sudo a2enmod ssl

sudo systemctl restart apache2
```

3. Отредактируйте файл конфигурации вашего сайта, обычно расположенный в `/etc/apache2/sites-available/`. Добавьте следующую конфигурацию:

```
<VirtualHost *:443>
ServerName yourdomain.com
DocumentRoot /var/www/html

SSLEngine on
SSLCertificateFile /path/to/yourdomain.crt
SSLCertificateKeyFile /path/to/yourdomain.key
SSLCertificateChainFile /path/to/chain.crt

# Дополнительные настройки безопасности
SSLProtocol all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
SSLHonorCipherOrder on
</VirtualHost>
```

4. Активируйте конфигурацию и перезапустите Apache:

```
sudo a2ensite your-ssl-site.conf
sudo systemctl restart apache2
```

Установка на Nginx.

1. Разместите файлы сертификата на сервере аналогично Apache.

2. Отредактируйте конфигурацию вашего сайта в Nginx (обычно в `/etc/nginx/sites-available/`):

```
server {
listen 443 ssl;
server_name yourdomain.com;

ssl_certificate /path/to/yourdomain.crt;
ssl_certificate_key /path/to/yourdomain.key;
```

```
    # Если у вас есть файл chain.crt, объедините его с
сертификатом
    # ssl_certificate /path/to/combined_cert.crt;


    # Оптимальные настройки SSL
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_prefer_server_ciphers on;
    ssl_ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-
RSA-AES128-GCM-SHA256;


    # Дополнительные настройки производительности
    ssl_session_cache shared:SSL:10m;
    ssl_session_timeout 10m;


    root /var/www/html;
    index index.html index.php;


    # Остальные настройки сайта
    }
```

3. Проверьте конфигурацию и перезапустите Nginx:

```
sudo nginx -t
sudo systemctl restart nginx
```

Для обоих серверов рекомендуется использовать современные протоколы (TLSv1.2 и TLSv1.3) и отключить устаревшие версии SSL и TLS, имеющие известные уязвимости. Также обратите внимание на правильные пути к файлам сертификатов — частая причина ошибок.

Настройка редиректов и решение распространённых проблем.

После установки SSL-сертификата необходимо настроить редирект с HTTP на HTTPS, чтобы все пользователи автоматически получали

защищенное соединение. Также рассмотрим распространенные проблемы и их решения.

Настройка редиректа с HTTP на HTTPS.

Для Apache добавьте следующую конфигурацию в файл HTTP-виртуального хоста (порт 80):

```
<VirtualHost *:80>
ServerName yourdomain.com
ServerAlias www.yourdomain.com

RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule ^(.*)$ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]
</VirtualHost>
```

Для Nginx используйте следующую конфигурацию:

```
server {
listen 80;
server_name yourdomain.com www.yourdomain.com;

return 301 https://$host$request_uri;
}
```

После настройки редиректа перезапустите веб-сервер.

Дополнительные оптимизации для Apache.

Добавьте в конфигурацию виртуального хоста следующие строки:

```
# Включение HTTP/2
Protocols h2 http/1.1
```

```
# OCSP Stapling
SSLUseStapling on
SSLStaplingCache "shmcb:logs/stapling-cache(150000)"
```

Для Nginx:

```
# Включение HTTP/2
listen 443 ssl http2;

# OCSP Stapling
ssl_stapling on;
ssl_stapling_verify on;
resolver 8.8.8.8 8.8.4.4 valid=300s;
resolver_timeout 5s;
```

Использование HSTS (HTTP Strict Transport Security) — это дополнительный уровень защиты, который указывает браузерам, что ваш сайт должен загружаться только по HTTPS:

Для Apache:

```
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
```

Для Nginx:

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

Примечание: включайте HSTS только после того, как убедитесь, что HTTPS работает корректно на всех разделах вашего сайта.

Проверка работы HTTPS и дополнительные меры безопасности.

После настройки HTTPS необходимо тщательно проверить корректность его работы и применить дополнительные меры для усиления безопасности вашего сайта.

Проверка правильной работы HTTPS.

Используйте следующие инструменты для комплексной проверки вашего HTTPS-соединения:

- SSL Labs Server Test (ssllabs.com/ssltest/) — Комплексная проверка конфигурации SSL/TLS вашего сайта с подробными рекомендациями.
- SSL Checker (sslshopper.com/ssl-checker.html) — Быстрая проверка сертификата и цепочки доверия.
- Why No Padlock (whynopadlock.com) — Помогает найти смешанный контент, который может блокироваться браузером.
- Chrome DevTools — Откройте консоль (F12), перейдите на вкладку "Security" для проверки проблем с HTTPS.

Основные моменты, на которые следует обратить внимание при проверке:

- Действительность сертификата и соответствие имени домена;
- Корректная установка цепочки сертификатов;
- Отсутствие смешанного контента;
- Работа редиректа с HTTP на HTTPS;
- Функциональность форм и интерактивных элементов;
- Проверка работы на всех основных браузерах.

Дополнительные меры безопасности.

После настройки базового HTTPS рекомендуется применить дополнительные меры для повышения безопасности:

1. Настройка Content Security Policy (CSP) — Защищает от XSS-атак, ограничивая источники загрузки ресурсов:

```
add_header Content-Security-Policy "default-src 'self';
script-src 'self' trusted-scripts.com;";
```

2. Настройка заголовка X-Content-Type-Options — Предотвращает MIME-sniffing:

```
add_header X-Content-Type-Options nosniff;
```

3. Настройка заголовка X-Frame-Options — Защита от clickjacking:

```
add_header X-Frame-Options SAMEORIGIN;
```

4. Настройка заголовка X-XSS-Protection — Дополнительная защита от XSS:

```
add_header X-XSS-Protection "1; mode=block";
```

5. Настройка заголовка Referrer-Policy — Контролирует передачу информации о реферере:

```
add_header Referrer-Policy strict-origin-when-cross-origin;
```

Автоматическое обновление сертификатов.

Если вы используете Let's Encrypt, настройте автоматическое обновление сертификатов с помощью cron-задачи:

```
0 3 * /usr/bin/certbot renew --quiet && /bin/systemctl restart nginx
```

Эта команда будет проверять и обновлять сертификаты ежедневно в 3 часа ночи и перезапускать веб-сервер при необходимости.

Мониторинг срока действия сертификата.

Установите систему оповещений о приближающемся истечении срока действия сертификата:

- Инструменты мониторинга: Nagios, Zabbix, Uptime Robot
- Специализированные сервисы: Cert Spotter, SSL Shopper
- Скрипт проверки с отправкой email-уведомлений

Регулярно проверяйте работу HTTPS после обновлений сервера или CMS, так как некоторые изменения могут нарушить настройки безопасности. Документируйте все изменения в конфигурации для облегчения поиска и устранения проблем в будущем.

Получение бесплатного ssl-сертификата с помощью утилиты mkcert.

mkcert — инструмент сильно упрощающий работу с SSL-сертификатами при локальной разработке. mkcert по сути выступает центром сертификации и позволяет выпускать доверенные SSL-сертификаты.

Поддерживаются Windows, Linux и macOS. В случае с Windows установка осуществляется через менеджеры пакетов Scoop или Chocolatey (рис. 59).

```
scoop install mkcert
```

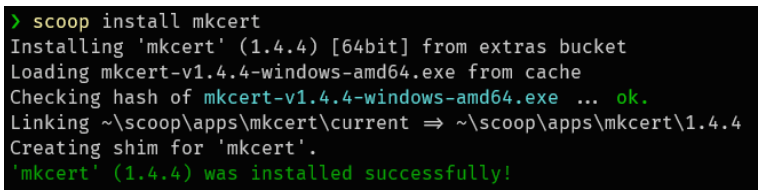


Рис. 59. Установка mkcert

После установки необходимо запустить команду инсталляции, во время выполнения которой mkcert создаст и добавит корневой сертификат в локальное хранилище доверенных центров сертификации. Поддерживается широкий список хранилищ (Windows, Linux, macOS, Firefox, Chrome и т. д.).

```
mkcert -install
```

Выпуск сертификата максимально прост, достаточно передать в mkcert список доменов, например:

```
mkcert predvoditelev.ru "*.predvoditelev.ru"
```

В результате будут созданы два файла:

- `predvoditelev.ru+1.pem` — сертификат;
- `predvoditelev.ru+1-key.pem` — секретный ключ.

Полный путь к генерируемым файлам можно предопределить, это бывает полезно при создании скриптов, автоматически создающих сертификаты с помощью mkcert. Пути задаются с помощью опций `-cert-file` и `-key-file`. Например:

```
mkcert -cert-file=./cert/cert.crt -key-
file=./cert/cert.key predvoditelev.ru
```

Работа с mkcert в WSL.

Для работы mkcert в Linux сначала нужно установить утилиту certutil. В случае с Ubuntu устанавливаем с помощью apt:

```
sudo apt install libnss3-tools
```

Далее устанавливаем mkcert. Есть несколько способов установки в Linux, все они описаны на странице инструмента. Можно просто скачать исполняемый файл:

```
curl -JLO
"https://dl.filippo.io/mkcert/latest?for=linux/amd64"
```

```
chmod +x mkcert-v*-linux-amd64
```

```
sudo cp mkcert-v*-linux-amd64 /usr/local/bin/mkcert
```

В WSL команду инсталляции не выполняем, так как под WSL мы будем только выпускать SSL-сертификаты. Но для того, чтобы сертификаты были доверенными в Windows, мы должны указать mkcert путь к корневому сертификату, который ранее был создан в Windows.

Смотрим, где находится корневой сертификат в Windows (рис. 60).

```
mkcert -CAROOT
```

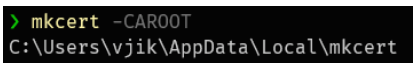


Рис. 60. Путь к установке сертификата

Далее нужно в WSL добавить переменную окружения `CAROOT` с путём к корневому сертификату. Я использую оболочку Bash, поэтому сделаю это с помощью файла `.bashrc`:

```
echo "export
CAROOT=/mnt/c/Users/vjik/AppData/Local/mkcert" >>
~/.bashrc
```

После добавления переменной нужно перезапустить Bash (можно просто перезайти в WSL).

Теперь можно выпускать сертификаты в WSL и в Windows они будут считаться доверенными.

Последовательность выполнения лабораторной работы

1. Получите бесплатный ssl-сертификат с помощью утилиты mkcert.
2. Установите его в систему.
3. Установите сертификат в ваш веб-сервер.
4. Показать результаты работы преподавателю.
5. Ответить на вопросы преподавателя.

Лабораторная работа №17 «Настройка программного файрволла для веб-приложения»

Теоретические сведения

Наиболее популярным файрволлом веб-приложений (Web Application Firewall) является бесплатный ModSecurity.

ModSecurity – это мощный инструмент, предназначенный для защиты веб-приложений от различных атак, таких как SQL-инъекции (SQLi), кросс-сайтовый скриптинг (XSS), инъекции кода и других видов вредоносных действий.

Он представляет из себя модуль для таких веб-серверов, как Apache, Nginx и других, и предоставляет механизм обнаружения и предотвращения атак на основе образцов или сигнатур. ModSecurity анализирует HTTP-трафик, проверяет его на наличие угроз безопасности и блокирует запросы, которые соответствуют известным атакам или паттернам поведения, установленным администратором.

Этот инструмент также предоставляет гибкие настройки, которые позволяют администраторам настраивать правила и фильтры для обеспечения безопасности веб-приложений, что делает его важным компонентом в области веб-безопасности.

Установка ModSecurity.

В данном разделе мы на наш тестовый сервер установим Apache и ModSecurity. Для начала подключитесь по SSH к вашему VDS с использованием пользователя, имеющего полномочия администратора, но не являющегося учётной записью root. Затем запустите обновление индексов пакетов:

```
$ sudo apt update
```

После чего, при помощи следующей команды установите веб-сервер Apache:

```
$ sudo apt install apache2
```

Поскольку ModSecurity доступен для установки из дефолтного репозитория Ubuntu, для его инсталляции также можно использовать утилиту `apt`:

```
$ sudo apt install libapache2-mod-security2
```

Чтобы запустить установленный модуль ModSecurity, выполните в командной строке:

```
$ sudo a2enmod security2
```

Теперь, для того, чтобы применить данное изменение, перезапустите веб-сервер:

```
$ sudo systemctl restart apache2
```

Первоначальная настройка ModSecurity.

На данном шаге перейдите в каталог `/etc/apache2/mods-enabled/`, после чего откройте для редактирования файл `security2.conf`:

```
$ cd /etc/apache2/mods-enabled/

$ sudo nano security2.conf
```

Здесь необходимо убедиться, что файл содержит следующую строку:

```
IncludeOptional /etc/modsecurity/*.conf
```

Данная строка описывает, где будут храниться конфигурационные файлы Modsecurity. Перейдите в данную директорию, после чего файл `modsecurity.conf-recommended` переименуйте в `modsecurity.conf`:

```
$ cd /etc/modsecurity/

$ sudo mv modsecurity.conf-recommended modsecurity.conf
```

Далее, откройте данный файл для редактирования:

```
$ sudo nano modsecurity.conf
```

В файле отыщите строку:

```
SecRuleEngine DetectionOnly
```

И приведите её к виду:

```
SecRuleEngine On
```

Теперь найдите строку, содержащую настройки уровней данных, которые будут отображаться в журналах аудита.

```
SecAuditLogParts ABDEFHIJZ
```

Эту строку также необходимо изменить. Оптимально настройки отображаемой в логах информации выглядят следующим образом:

```
SecAuditLogParts ABCEFHJKZ
```

По окончании редактирования сохраните конфигурационный файл и закройте его.

Затем перезапустите веб-сервер для того, чтобы внесённые изменения вступили в силу:

```
$ sudo systemctl restart apache2
```

Установка базового набора правил.

Как мы уже говорили выше, ModSecurity защищает ваши веб-приложения при помощи правил для обнаружения и блокировки вредоносных агентов. Базовый набор OWASP ModSecurity Core Rule Set (CRS) – это стандартный комплекс, используемых в ModSecurity правил. Он может быть интегрирован с проектом Honeypot и содержит правила, позволяющие остановить векторы командных атак, включая SQL-инъекции, межсайтовый скриптинг и многие другие, которые могут быть использованы для обнаружения ботов и идентификации используемых сканеров.

При установке ModSecurity из стандартного репозитория Ubuntu вместе с ним инсталлируется также пакет modsecurity-crs. Он, в свою очередь, включает в себя базовый набор правил OWASP версии 3.x. Загрузить актуальный релиз CRS OWASP можно с его страницы на GitHub. Для загрузки набора правил версии, например, 3.3.2 в директорию `/tmp/` используйте следующие команды:

```
$ cd /tmp/
```

```
$ wget
https://github.com/coreruleset/coreruleset/archive/refs
/tags/v3.3.2.tar.gz
```

Если в вашей системе не установлена утилита `wget`, то необходимо проинсталлировать её из стандартного репозитория Ubuntu:

```
$ sudo apt install wget
```

Затем разархивируйте загруженный файл:

```
$ tar xvf v3.3.2.tar.gz
```

Теперь для сохранения файлов CRS создайте каталог `modsecurity-crs`:

```
$ sudo mkdir /etc/apache2/modsecurity-crs/
```

После чего переместите в него содержимое распакованного архива:

```
$ sudo mv coreruleset-3.3.2/
/etc/apache2/modsecurity-crs/
```

На следующем шаге перейдите в директорию `coreruleset-3.3.2/` и создайте там файл конфигурации CRS с использованием файла-примера:

```
$ cd /etc/apache2/modsecurity-crs/coreruleset-3.3.2/
```

```
$ sudo mv crs-setup.conf.example crs-setup.conf
```

Далее откройте для редактирования файл `security2.conf` из директории `/etc/apache2/mods-enabled/`:

```
$ cd /etc/apache2/mods-enabled/
```

```
$ sudo nano security2.conf
```

В файле найдите и удалите следующую строку:

```
IncludeOptional /usr/share/modsecurity-crs/*.load
```

Вместо удалённой строки перед тегом `</IfModule>` добавьте следующий текст:

```
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.2/crs-setup.conf
```

```
IncludeOptional /etc/apache2/modsecurity-crs/coreruleset-3.3.2/rules/*.conf
```

Затем закройте файл с сохранением внесённых изменений. После чего протестируйте Apache командой (рис. 61):

```
$ sudo apache2ctl -t
```

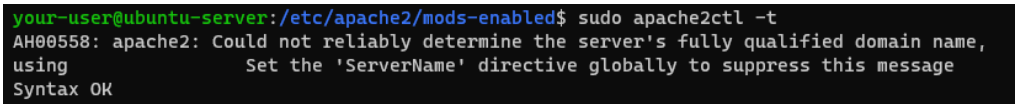


Рис. 61

И наконец, для применения новых настроек перезапустите веб-сервер:

```
$ sudo systemctl restart apache2
```

Проверка настроек ModSecurity.

Для того, чтобы протестировать работу ModSecurity, запустите простейшую атаку на свой сайт с использованием SQLi. Сделать это можно набрав в браузере следующий URL:

```
your-domain.host/?id=3 or 'a'='a'
```

Здесь, `your-domain.host` – доменное имя вашего сайта, вместо которого вы можете использовать его IP-адрес (рис. 62).

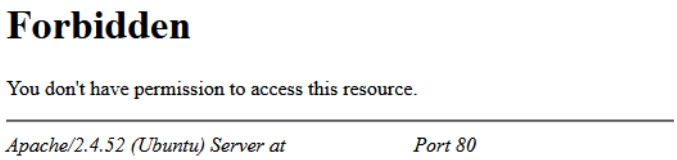


Рис. 62. Проверка работы ModSecurity

При проведении имитации атаки анализ журнала `/var/log/apache2/modsec_audit.log` показывает, что ModSecurity выявил и после чего заблокировал атаку применив OWASP CRS v3.3.2. Информацию об этом можно получить исходя из наличия соответствующих записей в разделе H (рис. 63).

Рис. 63. Анализ логов работы ModSecurity

Установка ModSecurity для защиты веб-сервера Apache на Ubuntu 22.04 представляет собой относительно простую процедуру, которая может быть выполнена путем установки соответствующих пакетов и настройки конфигурационных файлов. При необходимости вы можете самостоятельно изучить настройку правил и фильтров в ModSecurity, что позволит вам дополнительно усилить защиту сервера, адаптируя её под конкретные потребности ваших веб-приложений. Использование данного инструмента обеспечивает уровень безопасности, необходимый для защиты от вредоносных атак, делая его важным компонентом в области веб-безопасности для систем, работающих в том числе под управлением операционных систем семейства Ubuntu.

Последовательность выполнения лабораторной работы

1. Загрузить и установить модуль ModSecurity для Apache.
2. Загрузить и установить стандартный набор правил OWASP.
3. Протестировать работу модуль, имитировав атаку.
4. Показать результаты работы преподавателю.
5. Ответить на вопросы преподавателя.

2 ОБЩАЯ ХАРАКТЕРИСТИКА САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

Самостоятельная работа - целенаправленная, планируемая в рамках учебного плана деятельность студентов, которая осуществляется по заданию, при методическом руководстве и контроле преподавателя, но без его непосредственного участия. Самостоятельная работа студентов является одной из важнейших составляющих образовательного процесса.

В учебном процессе учебного заведения выделяют два вида самостоятельной работы: аудиторная и внеаудиторная.

Аудиторная самостоятельная работа выполняется на учебных занятиях под непосредственным руководством преподавателя и по его заданию.

Внеаудиторная — планируемая учебная, учебно-исследовательская, научно-исследовательская работа студентов, выполняемая во внеаудиторное время по заданию и при методическом руководстве преподавателя, но без его непосредственного участия.

Целью самостоятельной работы студентов является:

- систематизация и закрепление полученных теоретических знаний и практических умений;
- углубление и расширение теоретических знаний;
- формирование умений использовать нормативную, правовую, справочную документацию и специальную литературу;
- развитие познавательных способностей и активности студентов, творческой инициативы, самостоятельности, ответственности, организованности;
- формирование самостоятельности мышления, способностей к саморазвитию, совершенствованию и самоорганизации;
- формирование общих и профессиональных компетенций.

Самостоятельная работа студентов должна быть хорошо спланирована и организована. При планировании такой работы необходимо учитывать условия, обеспечивающие её успешное выполнение:

- чёткое определение преподавателем объёма и содержания самостоятельной работы;
- определение видов консультативной помощи;
- постановка цели самостоятельной работы и критерии её оценки;
- виды и формы контроля её выполнения.

Выполняя самостоятельную работу под контролем преподавателя, студент должен:

- освоить минимум знаний;
- планировать свою самостоятельную работу в соответствии разработанным графиком;
- выполнять самостоятельную работу и отчитываться по ее результатам в соответствии с графиком представления результатов, видами и сроками отчетности по самостоятельной работе студентов.

В процессе самостоятельной работы студент приобретает навыки самоорганизации, самоконтроля, самоуправления, саморефлексии и становится активным самостоятельным субъектом учебной деятельности.

Таким образом, самостоятельная работа студентов оказывает важное влияние на формирование личности будущего специалиста.

Самостоятельная работа студентов является обязательной для каждого студента, объем ее определяется учебным планом в соответствии с требованиями Государственных образовательных стандартов.

При изучении тем дисциплины студенты выполняют следующие виды самостоятельной работы:

- проработка конспектов занятий, учебных изданий и специальной технической литературы;
- составление конспекта, тематических схем, таблиц;
- подготовка к лабораторным работам и практическим занятиям с использованием методических рекомендаций преподавателя;
- оформление отчетов по лабораторным работам и практическим занятиям, подготовка к их защите;
- моделирование и решение производственных процессов и ситуационных задач;
- подготовка презентаций;
- работа с электронными ресурсами в сети Интернет;
- подготовка к семинару;
- подготовка к зачетам, экзаменам.

Технология организации самостоятельной работы студентов включает использование информационных и материально-технических ресурсов образовательного учреждения. Материально-техническое и информационно - техническое обеспечение самостоятельной работы студентов включает в себя:

- библиотеку с читальным залом, укомплектованную в соответствии с существующими нормами;
- учебно-методическую базу учебных кабинетов, лабораторий и методического центра;
- компьютерные классы с возможностью работы в Интернет;
- базы практики в соответствии с заключенными договорами;
- аудитории для консультационной деятельности;
- учебную и учебно-методическую литературу, разработанную с учетом увеличения доли самостоятельной работы студентов, и иные методические материалы.

Перед выполнением внеаудиторной самостоятельной работы преподаватель проводит инструктаж по выполнению задания, в котором указывает цель задания, его содержание, сроки выполнения, ориентировочный объем работы, основные требования к результатам работы, критерии оценки. Во время выполнения студентами внеаудиторной самостоятельной работы и при необходимости преподаватель может проводить консультации. Самостоятельная работа может осуществляться

индивидуально или группами студентов в зависимости от цели, объема, конкретной тематики самостоятельной работы, уровня сложности, уровня умений обучающихся.

3 МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Общие методические рекомендации студенту при изучении тем дисциплины.

Большая часть самостоятельной работы выполняется студентом вне учебных занятий при подготовке домашних заданий. Общие требования к выполнению этого вида самостоятельной работы заключаются в следующем:

- активно работать на уроке, усваиваю основную часть нового материала;
- если что-то непонятно, не стесняться задавать вопросы преподавателю;
- большое задание необходимо разбивать на части и работать над каждой из них в отдельности;
- выполняя домашнее задание, надо не просто думать, что надо сделать, а еще и решать, с помощью каких средств и приемов этого можно добиться;
- в процессе приготовления домашнего задания необходимо делать перерывы;
- готовиться к докладам, рефератам, защите курсовых работ и проектов, практических и лабораторных занятий надо заранее, равномерно распределяя нагрузку, а не оставлять такую ответственную работу на последний день;
- изучая заданный материал, сначала надо его понять, а уже потом запомнить;
- научиться находить интересующую нужную информацию с помощью компьютера;
- не стесняться обращаться за помощью к взрослым и однокурсникам;
- надо составлять план устного ответа и проверять себя;
- на письменном столе должно лежать только то, что необходимо для выполнения одного задания. После его завершения со стола убираются уже использованные материалы, и кладутся те учебные принадлежности, которые необходимы для выполнения следующего задания;
- нужно решить, в какой последовательности лучше выполнять задания и сколько времени понадобится на каждое из них;
- трудный материал урока лучше повторить в тот же день, чтобы сразу закрепить его и запомнить;
- читая учебник, надо задавать самому себе вопросы по тексту.

Подготовка тематических сообщений, докладов, рефератов

Реферат доклад, сообщение (от латинского refero - передаю, сообщаю) - краткое письменное изложение материала по определенной теме с целью привития студентам навыков самостоятельного поиска и анализа информации, формирования умения подбора и изучения литературных

источников, используя при этом дополнительную научную, методическую и периодическую литературу.

Тема реферата выбирается по желанию студента из списка, предлагаемого преподавателем. Тема может быть сформулирована студентом самостоятельно.

Выбранная тема согласовывается с преподавателем.

После выбора темы требуется:

- составить план реферата;
- подобрать необходимую информацию;
- изучить подобранную информацию;
- составить текст реферата.

План реферата должен включать в себя введение, основной текст и заключение. Во введении аргументируется актуальность выбранной темы, указываются цели и задачи исследования. В нем также отражается методика исследования и структура работы. Основная часть работы предполагает освещение материала в соответствии с планом. В заключении излагаются основные выводы и рекомендации по теме исследования.

Реферат оформляется согласно требованиям, установленным в учебном заведении. Он должен содержать: титульный лист, оглавление и список использованной литературы. На титульном листе указываются: название учебного заведения, название профессионального модуля, междисциплинарного курса, тема работы, курс, группа, фамилии, имена, отчества студента и руководителя работы, название города, в котором находится учебное заведение, год написания данной работы. Реферат может содержать приложения в форме схем, образцов документов и другие изображения в соответствии с темой исследования. Все страницы работы, включая оглавление и список литературы, нумеруются по порядку с титульного листа (на нем цифра не ставится) до последней страницы без пропусков и повторений. Введение, заключение, новые главы, список использованных источников и литературы должны начинаться с нового листа. Подбор литературы производится студентом из предложенного преподавателем списка литературы. Текст реферата необходимо набирать на компьютере на одной стороне листа. Размер левого поля 30 мм, правого - 15 мм, верхнего - 20 мм, нижнего - 20 мм. Шрифт - Times New Roman, размер - 14, межстрочный интервал - 1,5. Фразы, начинающиеся с новой строки, печатаются с абзацным отступом от начала строки (1,25 см). Реферат, выполненный небрежно, неразборчиво, без соблюдения требований по оформлению, возвращается студенту без проверки с указанием причин возврата на титульном листе.

Критерии оценки:

- знание и понимание проблемы;
- умение систематизировать и анализировать материал, четко и обоснованно формулировать выводы;
- «трудозатратность» (объем изученной литературы, добросовестное отношение к анализу проблемы);

- самостоятельность, способность к определению собственной позиции по проблеме и к практической адаптации материала, недопустимость плагиата;
- выполнение необходимых формальностей (точность в цитировании и указании источника текстового фрагмента, аккуратность оформления).

Проработка занятый, учебных изданий и специальной технической литературы

Работа с конспектом лекций по темам междисциплинарных курсов заключается в том, что студент после рассмотрения темы на учебных занятиях в период между очередными лекциями изучает материал конспекта. При этом непонятные положения конспекта необходимо выяснять у преподавателя на консультациях или при чтении основной и дополнительной литературы.

При работе с книгой необходимо научиться правильно ее читать, вести записи. Для подбора литературы в библиотеке используются алфавитный и систематический каталоги. Правильный подбор учебников рекомендуется преподавателем. Необходимая литература может быть также указана в методических разработках. Изучая материал по учебнику, следует переходить к следующему вопросу только после правильного уяснения предыдущего, описывая на бумаге все выкладки и определения (в том числе те, которые в учебнике опущены или на лекции даны для самостоятельного вывода). Полезно составлять опорные конспекты. При изучении материала по учебнику, полезно в тетради (на специально отведенных полях) дополнять конспект лекций, написанный на учебных занятиях. Там же следует отмечать вопросы, выделенные студентом для консультации с преподавателем. Выводы, полученные в результате изучения, рекомендуется в конспекте выделять, чтобы они при пропитывании записей лучше запоминались. Различают два вида чтения; первичное и вторичное. Первичное - эти внимательное, неторопливое чтение, при котором можно остановиться на трудных местах. После него не должно остаться ни одного непонятного слова. Содержание не всегда может быть понятно после первичного чтения. Задача вторичного чтения - полное усвоение смысла целого (по счету это чтение может быть и не вторым, а третьим или четвертым).

Чтение научного текста является частью познавательной деятельности. Ее цель - извлечение из текста необходимой информации. От того на сколько осознанна читающим собственная внутренняя установка при обращении к печатному слову (найти нужные сведения, усвоить информацию полностью или частично, критически проанализировать материал и т.п.) во многом зависит эффективность осуществляемого действия. Выделяют четыре основные установки в чтении научного текста:

- информационно-поисковая, задача которой - найти, выделить искомую информацию;

- усваивающая, при которой усилия читателя направлены на то, чтобы как можно полнее осознать и запомнить как сами сведения, излагаемые автором, так и всю логику его рассуждений;
- аналитико-критическая - читатель стремится критически осмыслить материал, проанализировав его, определив свое отношение к нему;
- творческая, создающая у читателя готовность в том или ином виде использовать суждения автора, ход его мыслей, результат наблюдения, разработанную методику, дополнить их, подвергнуть новой проверке.

Самостоятельная работа при чтении учебной литературы начинается с изучения конспекта материала, полученного при слушании лекций преподавателя. Полученную информацию необходимо осмыслить. При необходимости, в конспект лекций могут быть внесены схемы, эскизы рисунков, другая дополнительная информация.

Составление конспекта, тематических схем, таблиц

При изучении нового материала, как правило, составляется конспект. Конспект - изложение текста, которому присущи краткость, связность и последовательность. При этом максимально точно записываются формулы, определения, схемы, трудные для запоминания места.

При оформлении конспекта необходимо стремиться к емкости каждого предложения. Мысли автора книги следует излагать кратко, заботясь о стиле и выразительности написанного. Число дополнительных элементов конспекта должно быть логически обоснованным, записи должны распределяться в определенной последовательности, отвечающей логической структуре текста. Для уточнения и дополнения необходимо оставлять поля. Овладение навыками конспектирования требует от студента целеустремленности, повседневной самостоятельной работы.

Классификация конспектов:

- плановый конспект, для чего сначала нужно написать план текста, а затем на пункты плана делаются комментарии: свободно изложенный текст либо цитаты;
- обзорный конспект - краткое изложение данной темы с использованием нескольких источников;
- текстуальный конспект состоит из цитат одного текста;
- свободный конспект предполагает цитаты текста и собственные формулировки прочитанного текста;
- сложный - конспект, в котором отражается определенная тема или вопрос;
- хронологический конспект отражает последовательность событий;
- опорный конспект, в котором излагается информация в виде опорных знаков, слов, сигналов.

Методические рекомендации по составлению конспекта:

- определить цель написания конспекта;

- внимательно прочитать текст, уточнить в справочной литературе непонятные слова;
- выделить основные смысловые части текста;
- определить главное, составить план;
- кратко сформулировать основные положения текста, отметить аргументацию автора;
- составить текст конспекта, изложив информацию кратко и своими словами, четко следуя пунктам плана, записи следует вести четко, ясно;
- грамотно записывать цитаты, учитывая лаконичность, значимость мысли;
- в тексте конспекта желательно приводить не только тезисные положения, но и их доказательства.

При составлении тематических схем, таблиц необходимо внимательно прочитать текст соответствующий параграф учебника. Продумать «конструкцию» таблицы или схемы, расположение порядковых номеров, терминов, примеров и пояснений (и прочего). Начертить схему или таблицу и заполнить ее графы необходимым содержимым.

Подготовка к лабораторным работам и практическим занятиям, оформление отчетов по лабораторным работам и практическим занятиям, подготовка к их защите

Программы профессиональных модулей предусматривают выполнение практических и лабораторных занятий.

Лабораторное занятие - форма учебного занятия, ведущей дидактической целью которого является экспериментальное подтверждение и проверка существующих теоретических положений (законов, зависимостей), формирование учебных и профессиональных практических умений и навыков.

Практическое занятие - это одна из форм учебной работы, которая ориентирована на закрепление изученного теоретического материала, его более глубокое усвоение и формирование умения применять теоретические знания в практических целях. Особое внимание на практических занятиях уделяется выработке учебных или профессиональных навыков. Такие навыки формируются в процессе выполнения конкретных заданий - упражнений, задач - под руководством и контролем преподавателя.

Подготовка к практическим и лабораторным занятиям заключается в работе с конспектом лекций по данной теме, в изучении соответствующего раздела учебника или учебного пособия, в просмотре дополнительной литературы. Этапы подготовки к практическому или лабораторному занятию заключаются в следующем: освежить в памяти теоретические сведения, полученные на лекциях и в процессе самостоятельной работы, подобрать необходимую учебную и справочную литературу. Отобрать те материалы, которые позволят в полной мере реализовать цели и задачи предстоящей работы. Еще раз проверить соответствие отобранного материала. Студент

должен прийти на лабораторное или практическое занятие подготовленным по данной теме.

При выполнении заданий практического или лабораторного занятия студент должен быть ознакомлен преподавателем с целью и ходом выполнения задания и, по необходимости, с правилами техники безопасности. Если у студентов во время выполнения заданий возникают вопросы, то преподаватель консультирует студентов. Порядок выполнения того или иного задания излагается в инструкционных картах или рабочих тетрадях.

После проведения занятия студент представляет письменный отчет, который оформляется в соответствие с принятыми в образовательном учреждении правилами. Отчеты оформляются на листах писчей бумаги формата А4 или в специальных рабочих тетрадях, разработанных преподавателем. Содержание отчета указано в инструкционных картах или рабочих тетрадях.

При подготовке к защите практических и лабораторных занятий студент должен ответить на контрольные вопросы, указанные также в инструкционных картах или рабочих тетрадях, проштудировав при этом конспект лекций, учебную литературу.

Моделирование и решение производственных процессов и ситуационных задач

При изучении дисциплины очень часто студенту приходится сталкиваться с профессиональными задачами и ситуациями, которые необходимо решить самостоятельно, как во время аудиторной работы, так и во время внеаудиторной. При решении таких задач необходимо:

- провести анализ ситуации для определения проблемы в целом; представить ситуацию и себя в качестве действующего в ней лица; проанализировать ошибочные или правильные действия всех участников ситуации;
- определить проблемные узлы - возможные причины и прогнозируемые последствия развития данной ситуации;
- рассмотреть условное прогнозирование развития ситуации: определить окончательную гипотезу, представить обоснованный и доказательный прогноз вероятностного развития ситуации; предложить варианты действий, обоснованные теоретически и, по возможности, подкрепленные практическим личным опытом, опираясь на принципы профессиональной этики; определить способы и методы воздействия на предлагаемую ситуацию;
- сформулировать итоговые выводы, используя профессиональные термины, доказательства правильности своего решения.

Подготовка презентаций

Подготовка презентации позволит студенту логически выстроить изучаемый материал, систематизировать его, сформировать

коммуникативные компетенции. Материал презентации представляется в виде текста, схем, диаграмм, таблиц, которые призваны дополнить текстовую информацию или передать ее в более наглядном виде. Желательно избегать в презентации изображений, не несущих смысловой нагрузки, если они не являются частью стилевого оформления. Цвет графических изображений не должен резко контрастировать с общим стилевым оформлением слайдов, иллюстрации рекомендуется сопровождать пояснительным текстом.

Анимационные эффекты используются для привлечения внимания слушателей или для демонстрации динамики развития какого - либо процесса. В этих случаях использование анимации оправдано, но не стоит чрезмерно насыщать презентацию такими эффектами, иначе это вызовет негативную реакцию аудитории.

Звуковое сопровождение должно отражать суть или подчеркивать особенность темы слайда, презентации. Фоновая музыка не должна отвлекать внимание слушателей и заглушать слова докладчика.

Оптимальное количество слайдов, как правило, десять - пятнадцать. Для оформления слайдов презентации рекомендуется использовать несложные шаблоны, соблюдать единый стиль. Не рекомендуется на одном слайде использовать более трех цветов. Смену слайдов для управления презентацией докладчиком желательно устанавливать по щелчку без времени. Шрифт, выбираемый для презентации, должен обеспечивать читаемость информации на экране и соответствовать выбранному шаблону оформления. Не желательно использовать разные шрифты в одной презентации.

Алгоритм выстраивания презентации должен соответствовать логической структуре работы и отражать последовательность ее этапов. Независимо от алгоритма выстраивания презентации на первом слайде рекомендуется выносить следующие данные: полное наименование образовательной организации; тема презентации; фамилия, имя, отчество студента; специальность обучения; фамилия, имя, отчество руководителя. Последний слайд должен содержать фразу «Спасибо за внимание».

Работа с электронными ресурсами в сети Интернет

Для повышения эффективности самостоятельной работы студент должен учиться работать в поисковой системе сети Интернет, в электронно-библиотечной системе и использовать найденную информацию при подготовке к занятиям.

Интернет сегодня - правомерный источник научных статей, статистической и аналитической информации, и использование его наряду с книгами давно уже стало нормой. Однако, несмотря на то, что ресурсы Интернета позволяют достаточно быстро и эффективно осуществлять поиск необходимой информации, следует помнить о том, что эта информация может быть неточной или вовсе не соответствовать действительности. В связи с этим при поиске материала по заданной тематике следует обращать

внимание на научные труды признанных авторов, которые посоветовали вам преподаватели.

Поиск информации можно вести по автору, заглавию, виду издания, году издания или издательству. Также в сети Интернет доступна услуга по скачиванию методических указаний и учебных пособий, подбору необходимой учебной и научно - технической литературы.

Подготовка к семинару

Семинар — это особая форма учебно-теоретических занятий, которая, как правило, служит дополнением к лекционному курсу. Семинар обычно посвящен детальному изучению отдельной темы.

Этапы подготовки к семинару:

- проанализировать тему семинара, подумать о цели и основных проблемах, вынесенных на обсуждение;
- внимательно прочитать материал, данный преподавателем по этой теме на лекции;
- изучить рекомендованную литературу, делая при этом конспекты прочитанного или выписки, которые понадобятся при обсуждении на семинаре;
- постараться сформулировать свое мнение по каждому вопросу и аргументированно его обосновать;
- записать возникшие во время самостоятельной работы с учебниками и научной литературой вопросы, чтобы затем на семинаре получить на них ответы.

При подготовке к семинарским занятиям следует руководствоваться указаниями и рекомендациями преподавателя, использовать основную и дополнительную литературу из представленного им списка.

При подготовке доклада на семинарское занятие желательно заранее обсудить с преподавателем перечень используемой литературы, за день до семинарского занятия предупредить его о необходимых для представления материала технических средствах. Напечатанный текст доклада представить преподавателю на рецензию.

Подготовка к зачетам, экзаменам

Изучение выше перечисленных тем дисциплины завершается зачетами или экзаменами.

Подготовка к зачету или экзамену способствует закреплению, углублению и обобщению знаний, получаемых в процессе обучения, а также применению их к решению практических задач. Готовясь к зачету или экзамену, студент ликвидирует имеющиеся пробелы в знаниях, углубляет, систематизирует и упорядочивает свои знания. На зачете или экзамене студент демонстрирует то, что он приобрел в процессе обучения конкретным темам междисциплинарных курсов или модулям в целом.

Экзаменационная сессия - это серия экзаменов, установленных учебным планом. Между экзаменами, согласно графику их проведения,

дается интервал времени в несколько дней. Не следует думать, что их достаточно для успешной подготовки к экзаменам. В эти дни нужно систематизировать уже имеющиеся знания. На консультации перед экзаменом студентов познакомят с основными требованиями, ответят на возникшие у них вопросы. Поэтому посещение консультаций обязательно.

Требования к организации подготовки студента к экзаменам те же, что и при занятиях в течение семестра, но соблюдаться они должны более строго. Во-первых, очень важно соблюдение режима дня: сон не менее 8 часов в сутки, занятия должны заканчиваться не позднее, чем за 2-3 часа до сна.

Оптимальное время занятий - утренние и дневные часы. В перерывах между занятиями рекомендуются прогулки на свежем воздухе, неутомительные занятия спортом. Во-вторых, наличие хороших собственных конспектов лекций. Даже в том случае, если была пропущена какая- либо лекция, необходимо во время ее восстановить, обдумать, снять возникшие вопросы для того, чтобы запоминание материала было осознанным. В-третьих, при подготовке к зачету или экзамену у студента должен быть хороший учебник или конспект литературы, прочитанной по указанию преподавателя в течение семестра. Здесь можно эффективно использовать листы опорных конспектов. Вначале следует просмотреть весь материал по сдаваемой теме, отметить для себя трудные вопросы, обязательно в них разобраться. В заключение еще раз целесообразно повторить основные положения. Систематическая подготовка к занятиям в течение семестра позволит использовать время экзаменационной сессии для систематизации знаний.

Правила подготовки к экзамену:

- сориентироваться во всем материале и обязательно расположить его согласно экзаменационным вопросам или вопросам, обсуждаемым на семинарах, учебных занятиях. Эта работа может занять много времени, но все остальное - уже технические детали, главное - это ориентировка в материале;
- постараться максимально запомнить материал, переосмыслить его, рассмотреть альтернативные идеи;
- подготовить «шпаргалки», главный смысл которых систематизация и оптимизация знаний, однако пользоваться таким подспорьем не рекомендуется. Это очень сложная и важная для студента работа, более сложная и важная, чем простое поглощение массы учебной информации. Если студент самостоятельно подготовил такие «шпаргалки», то, скорее всего, он и экзамены сдавать будет более уверенно, так как у него уже сформирована общая ориентировка в сложном материале. Как это ни парадоксально, но использование «шпаргалок» часто позволяет отвечающему студенту лучше демонстрировать свои познания, точнее - ориентировку в знаниях, что намного важнее знания «запомненного» и «тут же забытого» после сдачи экзамена.

При ответе на экзамене студент сначала должен продемонстрировать преподавателю усвоенный по программе обучения материал, и лишь после этого высказать иную, желательно аргументированную точку зрения.

4 МЕТОДИКА ВЫПОЛНЕНИЯ ВНЕАУДИТОРНОЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Получить у преподавателя задание и необходимую литературу.
2. Найти предложенную литературу на образовательном портале или в библиотеке.
3. Изучить имеющуюся литературу в электронном или печатном виде, прочитать материалы лекций, практических и (или) семинарских занятий по теме.
4. Изучить методические рекомендации.
5. Оформить работу в тетради или на компьютере в соответствии с требованиями преподавателя.
6. Сдать самостоятельную работу преподавателю, предварительно ответив на вопросы для самоконтроля.

5 МЕТОДЫ КОНТРОЛЯ И ОЦЕНКА ВНЕАУДИТОРНОЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Контроль результатов самостоятельной работы проводится преподавателем одновременно с текущим и промежуточным контролем знаний обучающихся. Для контроля самостоятельной работы обучающегося используются разнообразные формы и методы: фронтальный, индивидуальный, выборочный, самоконтроль, защита презентации, участие в семинарском занятии, ответы на контрольные вопросы и т. д. При контроле результатов самостоятельной работы используются следующие критерии:

- уровень освоения обучающимся учебного материала;
- умение обучающегося использовать теоретические знания при выполнении заданий;
- обоснованность и чёткость изложения ответа;
- оформления материала в соответствии с требованиями.

Критерии оценки выполненной обучающимися работы:

оценка «5» - работа выполнена без ошибок; чисто, без исправлений; тема раскрыта полностью;

оценка «4» - работа выполнена с незначительными ошибками; тема раскрыта не полностью;

оценка «3» - работа выполнена со значительными ошибками; тема практически не раскрыта;

оценка «2» - работа не выполнена.

СПИСОК РЕКОМЕНДУЕМЫХ ИСТОЧНИКОВ

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 (дата обращения: 24.12.2025).
2. *Внуков, А. А.* Основы информационной безопасности: защита информации : учебное пособие для среднего профессионального образования / А. А. Внуков. — 3-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2024. — 161 с. — (Профессиональное образование). — ISBN 978-5-534-13948-8. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/542340 (дата обращения: 24.12.2025).
3. 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 (дата обращения: 24.12.2025).
4. *Стасышин, В. М.* Базы данных: технологии доступа : учебник для среднего профессионального образования / В. М. Стасышин, Т. Л. Стасышина. — 2-е изд., испр. и доп. — Москва : Издательство Юрайт, 2025. — 164 с. — (Профессиональное образование). — ISBN 978-5-534-09888-4. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/565153 (дата обращения: 24.12.2025).
5. *Щербак, А. В.* Поддержка и тестирование программных модулей : учебник для среднего профессионального образования / А. В. Щербак. — Москва : Издательство Юрайт, 2025. — 145 с. — (Профессиональное образование). — ISBN 978-5-534-19290-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/580603 (дата обращения: 24.12.2025).
6. *Карсян, А. Ж.* Цифровые технологии. Язык HTML : учеб.-метод. пособие. Ч. 1 / А. Ж. Карсян ; ФГБОУ ВО РГУПС. - Ростов н/Д : РГУПС, 2024. - 52 с
7. *Панасов, В. Л.* Разработка веб-страниц : учеб.-метод. пособие. Ч. 2 / В. Л. Панасов ; ФГБОУ ВПО РГУПС. - Ростов н/Д : [б. и.], 2013. - 23 с. : ил.
8. *Ломаш, Д. А.* Системы электронного документооборота и WEB-технологии на транспорте : учеб.-метод. пособие : в 4ч. Ч. 1. Разработка интерактивных веб-страниц на языке PHP с использованием СУРБД MySQL / Д. А. Ломаш ; ФГБОУ ВО РГУПС. - Ростов н/Д : [б. и.], 2016. - 53 с.

9. *Евдокимов, А. В.* Системы управления реляционными базами данных : учеб. пособие / А. В. Евдокимов, Н. М. Нечитайло ; ФГБОУ ВПО РГУПС. - Ростов н/Д : [б. и.], 2013. - 165 с.