|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | | | | | | |
| Институт информационных технологий | | | | | |
| Кафедра Инструментального и прикладного программного обеспечения (ИиППО)  **ПРАКТИЧЕСКАЯ РАБОТА №5** | | | | | |
| по дисциплине «Разработка серверных частей интернет-ресурсов» | | | | | |
| **Сессии. Файлы. БД.**  Студент группы ИКБО-20-19 | | | Московка А.А.  (подпись студента) | | |
|  | | |  | | |
| Руководитель практической работы | | | преподаватель Волков М.Ю. | | |
| Работа представлена | | «\_\_\_»\_\_\_\_\_\_\_2021 г. | | (подпись руководителя) | |
| Допущен к работе | | «\_\_\_»\_\_\_\_\_\_\_2021 г. | |  | |
|  | | Москва 2021 | |  | |

**ОГЛАВЛЕНИЕ**

Цель работы …………………………………………………………………………………... 3

Выполнение практической работы...................………………………………………….........4

Вывод............................................................................................................................................9

Ответы на вопросы……………………………………………………………………………..9

Список литературы……………………………………………………………………………17

**Задание.**

Предполагается выполнить апгрейд разрабатываемого в процессе первых 4 практических работ интернет-ресурса механизмами обработки сессий и согласования контента. Предлагается добавить следующую функциональность:

1. Хранение данных сессий в БД Redis.
2. Использование данных для согласования контента на уровне сервера для формирования контента пользователя с помощью (выбор по варианту). Требуется использовать хотя бы 3 параметра для формирования индивидуального контента, например, логин пользователя, тема (темная, светлая или для людей с цветовой слепотой) и рекомендуемый язык.
3. Загрузка файлов в формате pdf на сервер и хранение их (выбор по варианту), а также их выдача обратно пользователю по запросу.
   1. в файловой системе сервера
   2. в не реляционной базе данных (БД Redis).

Предполагается создание стабильной версии интернет-ресурса и сохранение предыдущей функциональности с практических работ 1-4.

* отчете требуется отразить и описать спецификацию реализованной функциональности, провести и отразить тестирование реализованной функциональности.

**Выполнение практической работы**

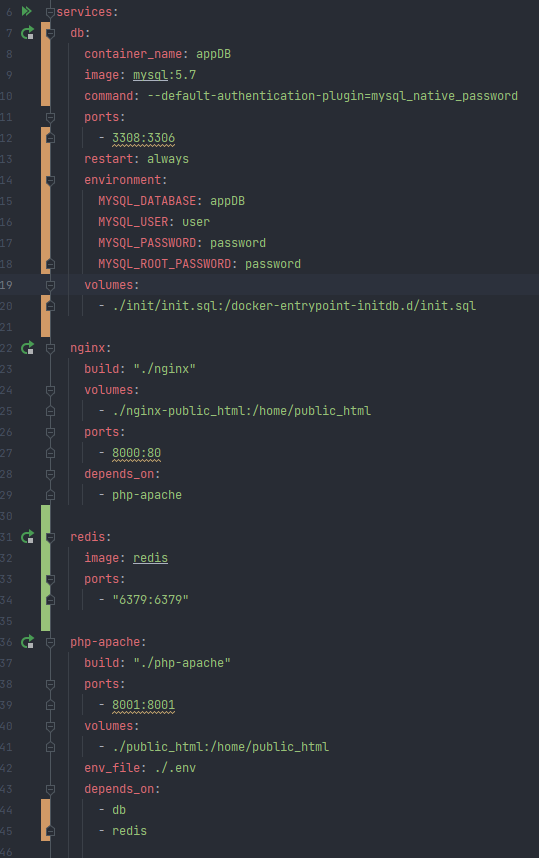
1. Для хранение данных сессий в БД Redis, мы будем менять docker-compose,dockerfile, перепишем хранение сессий на расширение php redis. 

Рисунок 1 – Скриншот добавления redis в файл docker-compose

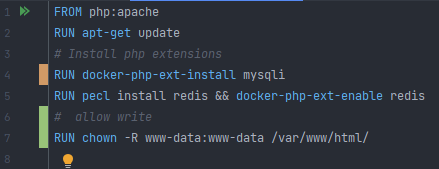


Рисунок 2 – Скриншот добавления расширения redis для php в файл Dockerfile

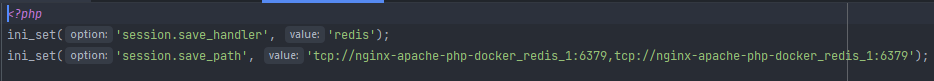


Рисунок 3 – Скриншот создания менеджера php, который будет переписывать данные сессий в Redis

1. Для формирования контента пользователю будет использован логин, выбор темной или светлой темы, и языковые предпочтения пользователя. Так же для проверки работоспособности будет выведена информация о количестве посещений странички.

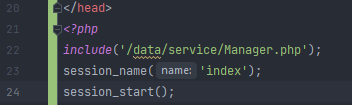


Рисунок 4 – Скриншот подключения модуля Manager.php и название сессии

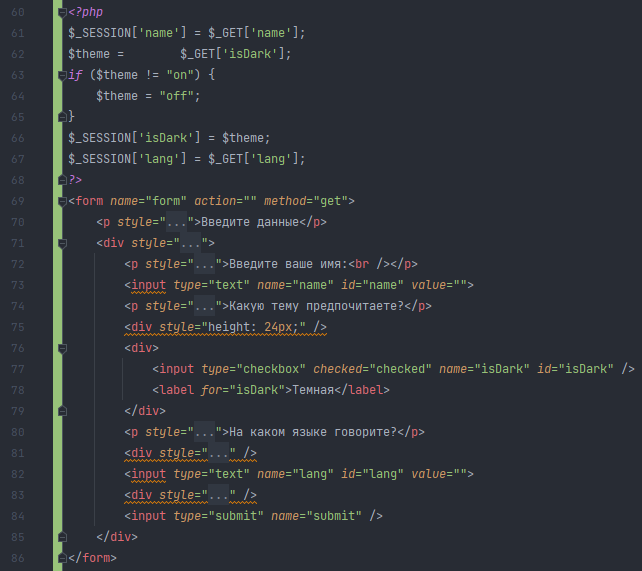


Рисунок 5 – Скриншот кода, в котором принимаются пользовательские данные и записываются в сессию

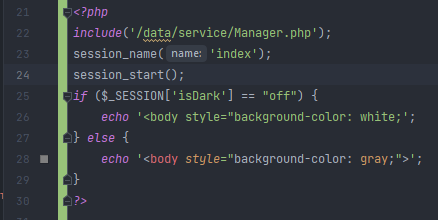


Рисунок 6 – Скриншот кода смены цвета темы

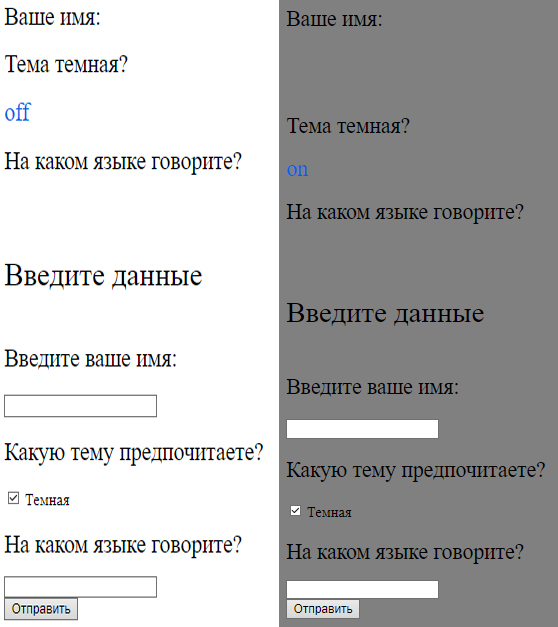


Рисунок 7 – Скриншот демонстрации смены цвета темы

1. Для загрузки файлов был создан еще один файл - upload.php. Если передать ему файл, он загрузит его на сервер в папку uploads. Модуль проверяет так же на тип файла – если будет загружен не pdf файл, пользователь получит ошибку. Для загрузки файла, пользователь может посмотреть доступные файлы на сервере, выбрать ему нужный и скачать, если файла не существует – пользователь ничего не получит.

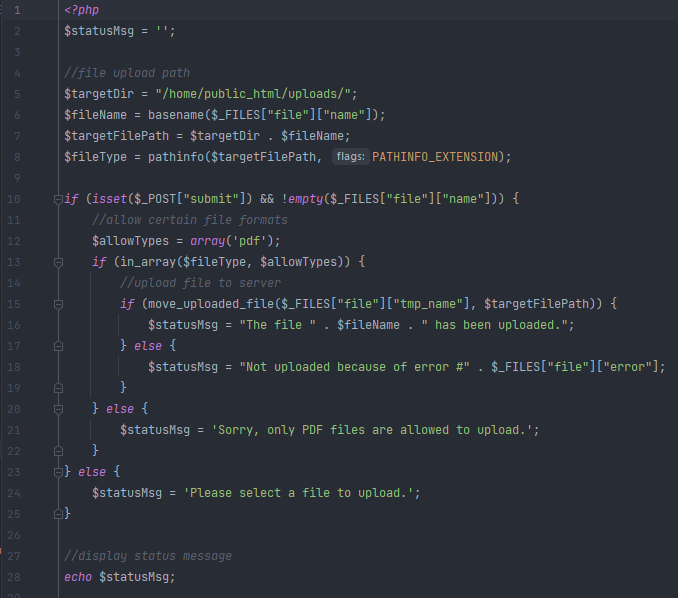


Рисунок 8 – Скриншот файла upload.php

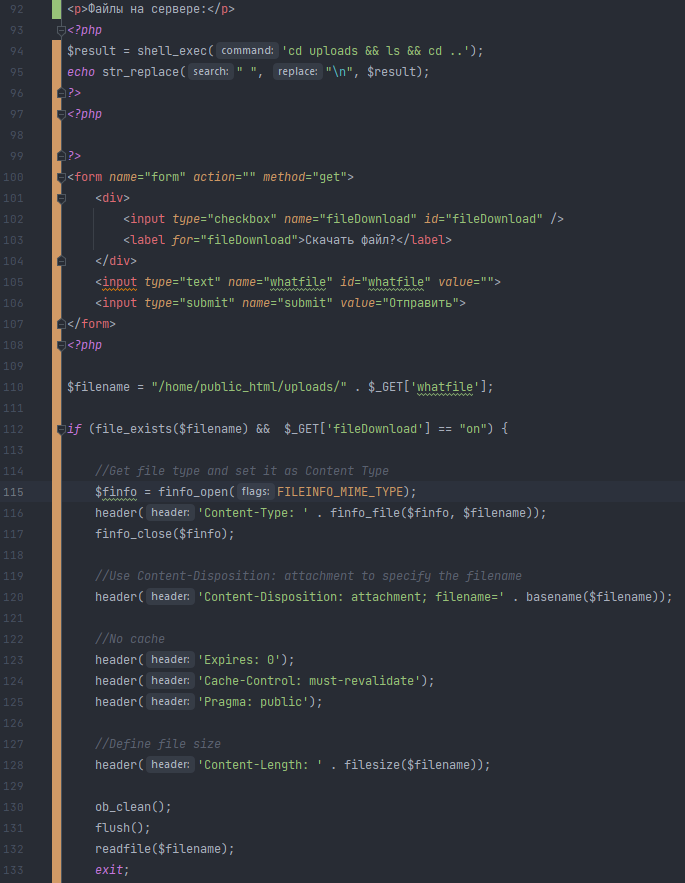


Рисунок 9 – Скриншот кода формы для загрузки файла pdf

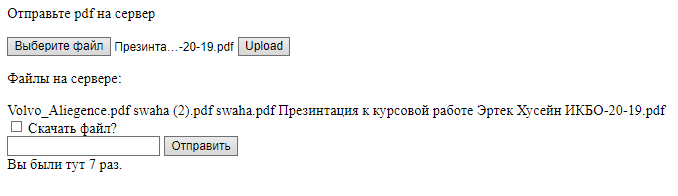


Рисунок 10 – Скриншот формы для загрузки файла pdf

**Вывод**

По итогу данной практической работы были получены навыки по сохранению сессий пользователя и выдаче персонального контента, был получен опыт работы с Redis.

**Ответы на вопросы:**

1. Сессия — это диалоговое состояние между клиентом и сервером, включающее информацию о предыдущих запросах клиента и ответах сервера. В силу того, что протокол и сами веб-сервера не обеспечивают сохранение состояния, то единственной возможностью для этого является передача всей необходимой информации о запросе в нём самом.
2. Куки, или cookies, — это служебные файлы, которые создаёт браузер для получения необходимой информации о пользователе.
3. Cookie используются веб-серверами для идентификации пользователей и хранения данных о них. К примеру, если вход на сайт осуществляется при помощи cookie, то, после ввода пользователем своих данных на странице входа, cookie позволяют серверу запомнить, что пользователь уже идентифицирован и ему разрешён доступ к соответствующим услугам и операциям.
4. Пример будет состоять из трёх файлов: index.php, authorize.php и secretplace.php. Файл index.php содержит форму, где пользователь введёт свой логин и пароль. Эта форма передаст данные файлу authorize.php, который в случае успешной авторизации допустит пользователя к файлу secretplace.php, а в противном случае выдаст сообщение об ошибке.
5. Использование cookie, шифрования, проверка браузера, срок действия сессии, привязка по IP-адресу.
6. Сессия как правило - это необходимые горячие данные пользователя, что нужно сохранить между запросами. БД - одно из самых медленных хранилищ этих данных. В файлах кстати тоже не стоит хранить. При большой нагрузке io будет подтормаживать. Длительный период сессии обычно не хранятся, вместо этого на клиент задается токен, по которому человек через много времени может автоматически авторизироваться.
7. Когда вы запускаете или продолжаете сеанс с помощью session\_start()$\_SESSION Когда выполнение скрипта заканчивается, данные сохраняются обратно в файл. Поэтому, когда вы устанавливаете переменную сеанса, она не сохраняется сразу. Конечно, вы можете заставить сеанс сохранять данные, вызывая session\_write\_close()

session\_set\_save\_handler() Требуется шесть аргументов, каждый из которых является обратным вызовом, который обрабатывает определенную стадию жизненного цикла сеанса. Они есть:

Открытие файла сеанса. Закрытие файла сеанса. Чтение данных сеанса. Запись данных сеанса. Уничтожение сессии. Сборка мусора из файла сессии и данных. Вы должны зарегистрировать функцию для каждого этапа жизненного цикла, иначе PHP выдаст предупреждение о том, что не может найти функцию.

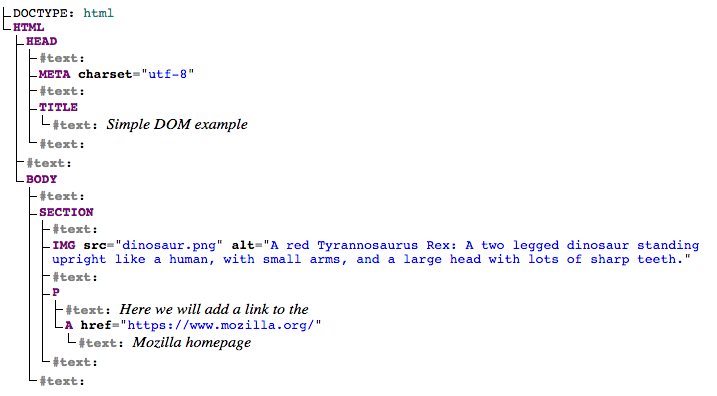
1. Скорее всего да, но тогда это может повлиять на безопасность сайта. Не будет возможности отслеживать действия пользователей, анализировать их для дальнейших действий, в случае подозрительного поведения. Также очень легко будет подвергнуться атакам со стороны взломщиков и тд.
2. session.save\_handler stringsession.save\_handler определяет имя обработчика, который используется для хранения и извлечения данных, связанных с сессией. По умолчанию имеет значение files. Следует обратить внимание, что некоторые модули могут зарегистрировать собственные обработчики (save\_handler). Текущие зарегистрированные обработчики отображаются в phpinfo().
3. allow\_url\_fopen bool. Данная директива включает поддержку обёрток URL (URL wrappers), которые позволяют работать с объектами URL как с обычными файлами. Обёртки, доступные по умолчанию, служат для работы с удалёнными файлами с использованием ftp или http протокола. Некоторые модули, например, zlib, могут регистрировать собственные обёртки.
4. Например, стандарт NTFS разработан с целью устранения недостатков, присущих более ранним версиям ФС. Впервые он был реализован в Windows NT в 1995 году, и в настоящее время является основной файловой системой для Windows. Система NTFS расширила допустимый предел размера файлов до шестнадцати гигабайт, поддерживает разделы диска до 16 Эб (эксабайт, 1018 байт). Использование системы шифрования Encryption File System (метод «прозрачного шифрования») осуществляет разграничение доступа к данным для различных пользователей, предотвращает несанкционированный доступ к содержимому файла. Файловая система позволяет использовать расширенные имена файлов, включая поддержку многоязычности в стандарте юникода UTF, в том числе в формате кириллицы. Встроенное приложение проверки жесткого диска или внешнего накопителя на ошибки файловой системы chkdsk повышает надежность работы харда, но отрицательно влияет на производительность.
5. Функция fopen() в PHP — это встроенная функция, которая используется для открытия файла или URL-адреса. В случае сбоя, она возвращает FALSE и выдаёт ошибку. Если вам нужно скрыть вывод ошибки добавьте символ '@' перед именем функции. Функция fclose() используется, чтобы закрыть открытый файл.
6. Для простого отображения всего содержимого файла идеально подходит функция readfile (). Для записи текстовых данных в файл существует две идентичные функции: int fputs ( int file, string string [, int length ]), int fwrite ( int file, string string [, int length ]).
7. string fgets ( int file, int length) – функция считывает строку длиною length. Функция fgets() используется для чтения одной строки из файла. Функция fgetc() используется для чтения одного символа из файла. ункция feof() полезна для цикла работающего с данным неизвестной длины.
8. Чтобы создать файл php, можно использовать функцию fopen() в режиме доступа «w» или «w+». Или функцию touch (). Она устанавливает время изменения файла. При отсутствии элемента с искомым именем он будет создан. Удалить файл можно с помощью функции unlink().
9. Для получения информации о файлах в php используется целый ряд функций:

bool fileexists (string filename) – проверяет, существует ли элемент;

int fileatime (string filename) – возвращает время последнего открытия;

int filesize (string filename) – возвращает байтовый размер файла;

string filetype (string filename) – тип файла.

1. DOM означает объектную модель документа. Это программный интерфейс, который позволяет нам создавать, изменять или удалять элементы из документа. Мы также можем добавлять события к этим элементам, чтобы сделать нашу страницу более динамичной. Модель DOM рассматривает документ HTML как дерево узлов.
2. 
3. JSON — это формат, который хранит структурированную информацию и в основном используется для передачи данных между сервером и клиентом. Файл JSON представляет собой более простую и лёгкую альтернативу расширению с аналогичными функциями XML (Extensive Markup Language).
4. Структуры JSON-данных очень похожи на массивы PHP. PHP имеет встроенные функции для кодирования и декодирования данных JSON. Это функции json\_encode() и json\_decode() соответственно. Обе функции работают только со строковыми данными в кодировке UTF-8.
5. Для этого можно воспользоваться сервисами валидации json файлов, которые могут проанализировать и указать на наличие ошибок в json файлах.
6. В php xml файлы создаются с помощью DOM функций, которые создают дерево объектов, в точности повторяющее дерево элементов XML документа. Так как DOM строит дерево всего документа, то и ресурсов он кушает немало (памяти и процессора он потребляет достаточно). Для создания XML документа используется класс – DomDocument.

Пример парсинга:

<?xml version='1.0'?> <worker> <name>Коля</name> <age>25</age> <salary>1000</salary> </worker>

Пример PHP:

$xml **=** simplexml\_load\_file(путь к файлу или урл);

echo $xml**->**name; //выведет 'Коля'

echo $xml**->**age; //выведет 25

echo $xml**->**salary; //выведет 1000

DOMDocument::saveXML — Сохраняет XML-дерево из внутреннего представления в виде строки.

1. Драйвера это библиотеки, позволяющие осуществлять взаимодействие между программой и СУБД/БД.

24. <?php

1. if (isset($\_POST["username"]) && isset($\_POST["userage"])) {
3. $conn = new mysqli("localhost", "root", "mypassword", "testdb2");
4. if($conn->connect\_error){
5. die("Ошибка: " . $conn->connect\_error);
6. }
7. $name = $conn->real\_escape\_string($\_POST["username"]);
8. $age = $conn->real\_escape\_string($\_POST["userage"]);
9. $sql = "INSERT INTO Users (name, age) VALUES ('$name', $age)";
10. if($conn->query($sql)){
11. echo "Данные успешно добавлены";
12. } else{
13. echo "Ошибка: " . $conn->error;
14. }
15. $conn->close();
16. }
17. ?>

25. Идея постоянных подключений состоит в том, чтобы соединение между клиентским процессом и базой данных можно было использовать повторно, особенно когда требуется создавать и закрывать соединения множество раз. Это бы позволило снизить накладные расходы на создание новых подключений каждый раз, когда они требуются, за счёт использования существующих кешированных подключений, свободных для повторного использования. При использовании постоянных соединений можно столкнуться с проблемой, которая заключается в том, что клиенты могут оставлять такие подключения в непредсказуемом состоянии. Например, клиент ставит блокировку на таблицу, а затем аварийно завершает работу. То есть блокировка снята не будет. Новый клиентский процесс, использующий это подключение повторно, получит его как есть, и вынужден будет провести какую-то очистку подключения, прежде чем начать его использовать. Соответственно, в задачи программиста входит ещё и проверка подобных ситуаций и внедрение кода, осуществляющего такую очистку. Тем не менее, в mysqli эта проблема решена. В модуле есть встроенный функционал, осуществляющий очистку соединений и переводящий их в состояние пригодное для использования. Код очистки, реализованный в mysqli включает следующие операции:

Откат активных транзакций

Закрытие и удаление временных таблиц

Снятие блокировки с таблиц

Сброс переменных сессии

Закрытие подготовленных запросов (всегда происходит в PHP)

Закрытие обработчиков

Снятие блокировок, установленных функцией GET\_LOCK()

Это позволяет быть уверенным в том, что возвращённые из пула соединения готовы к использованию в клиентских процессах. Модуль mysqli делает очистку соединений автоматически путём вызова C-API функции mysql\_change\_user().

26. Главные особенности MongoDB:

Это кроссплатформенная документоориентированная база данных NoSQL с открытым исходным кодом.

Она не требует описания схемы таблиц, как в реляционных БД. Данные хранятся в виде коллекций и документов.

Между коллекциями нет сложных соединений типа JOIN, как между таблицами реляционных БД. Обычно соединение производится при сохранении данных путем объединения документов.

Данные хранятся в формате BSON (бинарные JSON-подобные документы).

У коллекций не обязательно должна быть схожая структура. У одного документа может быть один набор полей, в то время как у другого документа — совершенно другой (как тип, так и количество полей).

В одном документе могут быть поля разных типов данных, данные не нужно приводить к одному типу. Основное преимущество MongoDB заключается в том, что она может хранить любые данные, но эти данные должны быть в формате JSON.

27. Для добавления в коллекцию могут использоваться три ее метода:

insertOne(): добавляет один документ

insertMany(): добавляет несколько документов

insert(): может добавлять как один, так и несколько документов

Пример добавления:

db.users.insertOne({"name": "Tom", "age": 28, languages: ["english", "spanish"]});

28. Для получения всех записей в коллекции используйте метод find() экземпляра коллекции. Обновление записи происходит с помощью метода updateOne(), который принимает условие для записей, которые необходимо обновить, и объект со свойством $set, в значении которого указываются, какие именно поля нужно изменить. За удаление записей отвечают методы deleteOne() и deleteMany().

29. Для получения данных в MySQL применяется команда SELECT. При выполнении команды SELECT метод query() объекта mysqli возвращает набор полученных строк, который мы можем перебрать с помощью цикла.

30. Функция find() позволяет возвратить несколько документов. Еще одна функция findOne() работает похожим образом, только возвращает один документ. Например:

db.users.find({name: "Tom", age: 32})

Для подсчета выборки можно использовать следующее:

db.collection.count(query, options)

Для того, чтобы извлечь лишь одно значение из полученной выборки можно воспользоваться методом findOne(). Однако, к методу findOne() прибегают чаще в тех ситуациях, когда ожидается, что результирующая коллекция будет содержать лишь один документ. В тех же случаях, когда следует ограничить выборку несколькими документами предпочитают использовать метод limit(), который принимает в качестве аргумента количество извлекаемых документов.

**Список литературы:**

1. Docker-compose // doka URL: https://doka.guide/tools/docker-compose/ (дата обращения: 11.08.2021).
2. Dockerfile // Doka URL: https://doka.guide/tools/dockerfile/ (дата обращения: 11.08.2021).
3. Карл Сегуин - Маленькая книга о Redis URL: https://3s.group/upload/iblock/890/890258c1a79e7154773b210ea701b184.pdf (дата обращения: 11.08.2021).
4. Модуль mysqli и постоянные соединения // php URL: https://www.php.net/manual/ru/mysqli.persistconns.php (дата обращения: 11.08.2021).
5. Руководство по PHP, Справочник функций, Обработка XML, DOM // php URL: https://www.php.net/manual/ru/domdocument.savexml.php (дата обращения: 11.08.2021).
6. Руководство по PHP, Справочник языка, Предопределённые переменные// php URL: https://www.php.net/manual/ru/reserved.variables.session.php (дата обращения: 11.08.2021).