rs

|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №4**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Студент группы** ИКБО-30-20 Бессмертных И. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Принял** Благирев М.М.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Москва 2022

ОГЛАВЛЕНИЕ

[Цель работы 3](#_Toc118206944)

[Задание 3](#_Toc118206945)

[Выполнение работы 3](#_Toc118206946)

[Выводы 8](#_Toc118206947)

[Ответы на вопросы 10](#_Toc118206948)

[Список информационных источников: 15](#_Toc118206949)

# Цель работы

Реализация взаимодействия клиента и сервера с использованием технологии API.

# Задание

Предполагается реализация интерфейса прикладного программирования для доступа к некоторым данным по варианту. Предполагается реализация серверной части обработки запросов и тестирование данного интерфейса с использованием программы Postman. Для реализации данного сервиса предлагается использовать серверную конфигурацию, модернизированную в течение первых трех практических работ. Важной частью данной практической работы является сохранение функциональности, реализованной в практической работе №3. То есть интерфейс предлагается создать уже в существующем веб-приложении. Также предполагается использование темы практической работы №3 для продолжения модернизирования собственной системы. Изменение темы согласовывается отдельно с преподавателем. Хранение данных предполагается уже в существующей базе данных. Технические требования к реализации интерфейса:

1. Доступ как минимум к 2 независимым сущностям.
2. Реализация как минимум операций группы CRUD (создание, чтение, обновление, удаление). Приветствуется реализация дополнительной функциональности.
3. Тестирование всех функциональных возможностей созданного интерфейса с использованием программы Postman.

# Выполнение работы

Тема практической работы была взята из работы №3. Создадим файл cars.php, который будет выполнять функционал API.

Работа запросов была протестирована с использованием ПО Postman. Результаты представлены на рисунках 1–5.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 - Все машины

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 - Добавление машины

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 - Получение машины по ID

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 – Редактирование машины

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 - Удаление машины

# Выводы

В ходе данной практической работы, были получены навыки в создании REST-full API реализующий CRUD на языке программирование PHP.

**Ссылка на Git-репозиторий:**

https://github.com/Ivanbessmertnykh/Data-Servers

# Ответы на вопросы

1. **Что такое HTTP-запрос?** HTTP сообщения — это обмен данными между сервером и клиентом. Есть два типа сообщений: запросы, отправляемые клиентом, чтобы инициировать реакцию со стороны сервера, и ответы от сервера.

2. **Опишите существующие HTTP-запросы.** Для разграничения действий с ресурсами на уровне HTTP-методов и были придуманы следующие варианты:

GET — получение ресурса

POST — создание ресурса

PUT — обновление ресурса

DELETE — удаление ресурса

3. **Опишите обработку запроса на PHP. Что нужно использовать, как**

**вычленить параметры запроса?**  Для обработки запроса достаточно создать файл-обработчик, который будет служить эндпоинтом, внутри для доступа к параметрам можно использовать предопределенные массивы \_GET и \_POST.

4. **Опишите создание HTML-форм на PHP.** Чтобы создать форму используется тег с двумя атрибутами, который можно вывести на страницу используя функцию echo: атрибут action. С его помощью указывается адрес, на который отправятся введенные на форме данные, атрибут method. С его помощью указывается HTTP-метод отправки формы.

5. **Что такое API?** API – Application Programming Interface, что значит программный интерфейс приложения. В контексте API слово «приложение» относится к любому ПО с определенной функцией. Интерфейс можно рассматривать как сервисный контракт между двумя приложениями. Этот контракт определяет, как они взаимодействуют друг с другом, используя запросы и ответы.

6. **Опишите API как средство интеграции приложений.** Интеграция API – это соединение между двумя или более приложениями через их API (интерфейсы прикладного программирования), которые позволяют системам обмениваться источниками данных. Интеграция API позволяет управлять процессами во многих секторах и уровнях организации, обеспечивая синхронизацию данных, повышая производительность и увеличивая прибыль.

7. **Что такое Web API?** Web API или Web Service API –это интерфейс обработки приложений между веб-сервером и веб-браузером. Все веб-сервисы являются API, но не все API являются веб-сервисами. REST API – это особый тип Web API, в котором используется стандартный архитектурный стиль, описанный выше.

8. **Приведите пример API.**  Примером API может служить данная работа.

9. **Что такое REST?** REST — архитектурный стиль взаимодействия компонентов распределённого приложения в сети. Другими словами, REST — это набор правил о том, как программисту организовать написание кода серверного приложения, чтобы все системы легко обменивались данными и приложение можно было масштабировать.

10. **Как организована передача данных в архитектуре REST?** Передача данных осуществляется через запросы, входящие параметры могут передаваться через url, заголовки или тело (например json объекты), запросы также возвращают значения от API. Передача данных обычно осуществляется через протоколы http и https, но могут быть использованны и другие.

11. **Как организована работа REST?** Данные (такие как изображения, видео и текст) воплощают ресурсы в REST. Клиент посещает URL-адрес и отправляет серверу запрос, чтобы получить ответ.

12. **Что такое SOAP?** SOAP — протокол обмена структурированными сообщениями в распределённой вычислительной среде. Первоначально SOAP предназначался в основном для реализации удалённого вызова процедур. Сейчас протокол используется для обмена произвольными сообщениями в формате XML, а не только для вызова процедур.

13. **Чем SOAP отличается от REST?** REST — это архитектурный стиль. SOAP — это формат обмена сообщениями. На верхнем уровне SOAP ограничивает структуры ваших сообщений, тогда как REST — это архитектурный подход, ориентированный на использование HTTP в качестве транспортного протокола.

14. **Для чего нужен SOAP-процессор?**  SOAP-процессор, совместимый с v1.1, генерирует ошибку при получении сообщения, содержащего пространство имен конверта v1.2. SOAP-процессор, совместимый с v1.2, генерирует ошибку VersionMismatch, если он получает сообщение, которое не включает пространство имен конверта v1.2. Каждое сообщение SOAP имеет корневой элемент Envelope.

15. **Опишите общую структуру SOAP-сообщения.** Сообщение SOAP выглядит так:

* Envelope — корневой элемент, который определяет сообщение и пространство имен, использованное в документе
* Header — содержит атрибуты сообщения, например: информация о безопасности или о сетевой маршрутизации
* Body — содержит сообщение, которым обмениваются приложения
* Fault — необязательный элемент, который предоставляет информацию об ошибках, которые произошли при обработке сообщений.

16. **Что такое и что содержит Конверт (SOAP Envelope)?** Envelope — корневой элемент, который определяет сообщение и пространство имен, использованное в документе. Содержится в сообщении.

17. **Что такое и что содержит Заголовок SOAP (SOAP Header)?** Header — содержит атрибуты сообщения, например: информация о безопасности или о сетевой маршрутизации. Содержится в сообщении

18. **Что такое и что содержит Тело SOAP (SOAP Body)?** Body — содержит сообщение, которым обмениваются приложения. Содержится в сообщении.

19. **Опишите SOAP-сообщение с вложением.**  Можно посылать вложением различные форматы: PDF, изображения или другие двоичные данные. Сообщения SOAP работают вместе с расширением MIME, в котором предусмотрено multipart/related

20. **Что такое graphql?** GraphQL — это язык запросов для API -интерфейсов и среда, в которой они выполняются. С помощью GraphQL можно получить данные из API и передать их в приложение (от сервера к клиенту). Официальная документация GraphQL есть только на английском языке, на русский язык пока ещё не переведена. GraphQL разработали в 2012 году как альтернативу REST.

21. **Что такое Распознаватели (resolvers) в graphql?** Resolver или распознаватель — функция, которая возвращает данные для определённого поля. Resolver’ы возвращают данные того типа, который определён в схеме. Распознаватели могут быть асинхронными. С их помощью можно получать данные из REST API, базы данных или другого источника.

22. **Из чего состоит экосистема graphql, что нужно, чтобы использовать**

**данную технологию?** Необходимо всего два компонента чтобы начать: сервер GraphQL для обработки запросов к API и клиент GraphQL, который будет подключаться к эндпоинтам.

23. .**Что такое валидация данных и для чего она нужна?**  Валидация данных — это процесс проверки данных различных типов по критериям корректности и полезности для конкретного применения. Валидация данных проводится, как правило, после выполнения операций ETL и для подтверждения корректности результатов работы моделей машинного обучения.

24. **Где и когда выполнять валидацию данных?**  Удобно проверять данные в том месте, где они попадают в программу из внешнего мира. После такой проверки можно быть уверенным, что в программу попадают правильные данные и в дальнейшем они могут использоваться без дополнительных проверок. Это может быть пользовательский интерфейс, через который человек вводит данные. Это может быть файл, содержащий настройки программы или данные, которые программа должна обработать. Это может быть база данных, в которую информация может попадать из других программ. Это может быть сетевой протокол обмена данными с другими программами. Наконец, это может быть программный интерфейс, который использует другая программа, вызывая некоторые функции/процедуры и передавая в них параметры.

25. **Как выполнять валидацию данных?** Валидация данных может осуществляться следующими методами: посимвольная проверка, проверка отдельных значений, совокупность входных значений, проверка состояния системы после обработки данных.

26. **Приведите пример с поэтапной валидацией данных**. Валидацию данных можно и нужно выполнять в несколько этапов, усложняя проверки. Сначала, по мере ввода, следим за тем, чтобы данные не содержали недопустимых символов. Например, для числового поля пользователю может быть запрещён ввод нецифровых символов. После того, как ввод завершён, можно проверить всё значение целиком. Для введённого числа могут быть какие-то ограничения, например, оно не должно превышать определённого максимального допустимого значения. Если наше числовое поле представляет собой возраст, оно должно находиться в пределах от 0 до, скажем, 120. Когда заполнены все поля, можно проверить, согласованы ли введённые значения друг с другом. Например, если в форме кроме поля для указания возраста есть поле для ввода номера паспорта, приложение может проверить, что при заполнении номера паспорта возраст должен быть не менее 14 лет. Наконец, если всё введено корректно, можно попытаться начать обработку, выполняя проверки по ходу дела, а также в самом конце, и если что-то пошло не так, выполнить откат к исходному состоянию. Ну и, конечно же, проверки на следующем уровне могут 14 подстраховывать проверки предыдущих уровней. Скажем, для вебприложений обязательной является проверка данных, пришедших на сервер в HTTP-запросе, независимо от того, выполнялась ли перед этим предварительная валидация в браузере или нет. Причина этого в том, что проверку на клиентской стороне можно обойти. Для других видов приложений обойти проверки не так просто, но иногда тоже вполне возможно, как показано в примере чуть ниже.

27. **Что такое запрос и мутация в graphql и чем они отличаются?** Запрос является интерактивным. Это означает, что Вы можете изменить его, как Вам нравится, и увидеть новый результат. Попробуйте добавить поле appearsIn в объект hero в запросе, и увидите новый результат. В запросе указываются те данные, которые необходимо получить, возможно указание параметров. Запрос GraphQL используется для чтения или извлечения значений, в то время как мутация используется для записи или публикации значений

# Список информационных источников:

1. Документация по языку PHP, электронный ресурс URL: https://php.net (дата обращения: 11.09.2022) – Текст: электронный;
2. Документация по системе Docker, электронный ресурс URL: https://docs.docker.com (дата обращения: 11.09.2022) – Текст: электронный.
3. Документация по HTML, CSS, JavaScript, электронный ресурс URL: https://developer.mozilla.org (дата обращения 11.09.2022) – Текст: электронный.