|  |  |
| --- | --- |
| **Описание: E:\!Кафедра ПИКС\Логотип БГУИР\Символика.jpg** | **Описание: E:\!Кафедра ПИКС\Логотип ПИКС\17 мая 2013\Логотип ПИКС_3.jpg** |

**ВОПРОСЫ К ЭКЗАМЕНУ**

**по дисциплине**

**«РАЗРАБОТКА WEB-СЕРВИСОВ»**

**Весенний семестр 2024-2025 учебного года**

**Специальность 1-39 03 02 «Программируемые мобильные системы)»**

**(группа 213871)**

1. Что такое веб-сервисы и как они отличаются от традиционных веб-приложений?
2. Какие основные преимущества предоставляют веб-сервисы в контексте современной веб-разработки?
3. Какие типы веб-сервисов существуют, и как они используются в современных приложениях?
4. Какова роль веб-сервисов в архитектуре микросервисов, и какие проблемы они помогают решать?
5. Что такое REST, и какие принципы лежат в его основе?
6. Какие ограничения накладывает REST на архитектуру веб-сервисов?
7. В чем разница между RESTful и SOAP-сервисами?
8. Какие преимущества и недостатки использования RESTful архитектуры в сравнении с другими подходами?
9. Что представляют собой ресурсы в контексте RESTful API, и как они связаны с сущностями приложения?
10. Какие HTTP методы используются для взаимодействия с ресурсами, и какие операции они представляют?
11. Какие принципы лежат в основе проектирования URI в RESTful API, и как они способствуют удобству и эффективности работы с API?
12. Какие соглашения следует соблюдать при проектировании RESTful API для обеспечения его гибкости, расширяемости и совместимости?
13. Какие инструменты используются для моделирования и документирования RESTful API, и какова их роль в процессе разработки? Какие функциональные возможности обычно предоставляют такие инструменты?
14. Какие проблемы могут возникнуть при отсутствии или недостаточной документации API, и как их можно избежать или решить?
15. Какие основные HTTP-методы используются для взаимодействия с веб-серверами, и для чего они предназначены?
16. Какие операции выполняют методы GET, POST, PUT, PATCH и DELETE? Какие могут быть практические примеры использования каждого из этих методов в контексте веб-разработки?
17. Что представляют собой параметры запросов и как они передаются в HTTP-запросах?
18. Какие типы параметров могут быть переданы через query parameters, headers и body запроса? Какие сценарии использования подразумевают передачу параметров через каждый из этих механизмов?
19. Каким образом клиент и сервер могут взаимодействовать с параметрами запросов для обеспечения эффективной коммуникации?
20. Чем отличаются JSON и XML как форматы обмена данными, и в каких сферах их используют?
21. Каковы основные преимущества и недостатки JSON и XML?
22. Какие типы данных могут быть представлены в JSON и XML, и какие синтаксические правила им следует придерживаться?
23. Какие библиотеки и инструменты доступны для обработки и генерации JSON и XML данных в различных языках программирования?
24. Какие факторы влияют на выбор формата данных для веб-сервисов?
25. В чем преимущества использования JSON перед XML и наоборот?
26. Какие альтернативные форматы данных могут быть использованы в современных веб-сервисах, и в каких сценариях они оправданы?
27. Какие основные критерии следует учитывать при выборе формата данных для конкретного веб-сервиса?
28. Какие методы аутентификации используются в современных веб-сервисах, и в чем заключаются их отличия?
29. Что представляет собой протокол OAuth и какие роли в нем играют аутентификация и авторизация?
30. Чем отличается использование JWT от традиционных методов аутентификации, и какие преимущества он предоставляет?
31. Что представляет собой механизм контроля доступа к ресурсам, и какие принципы лежат в его основе?
32. Какие могут быть стратегии реализации ролевой модели доступа к ресурсам в веб-сервисах?
33. Какие инструменты и библиотеки доступны для управления авторизацией и контролем доступа в различных средах разработки?
34. Какие стандартные HTTP-статусы ошибок существуют, и какие виды ошибок они представляют?
35. Какие группы статусов обозначают успешные операции, а какие - ошибки?
36. Какие информационные, успешные, перенаправляющие, ошибочные и серверные статусы могут быть возвращены сервером?
37. Как использование соответствующих HTTP-статусов помогает клиентским приложениям обрабатывать ошибки и адекватно реагировать на них?
38. Какие методы обработки ошибок используются в RESTful API, и какие принципы им следует придерживаться?
39. Какие стандартные практики возвращения ошибок в JSON формате применяются в современных веб-сервисах?
40. В чем заключается разница между возвращением информации об ошибках через HTTP статусы и через тело ответа?
41. Какие могут быть стратегии обработки и логирования ошибок на стороне сервера, чтобы обеспечить надежность и безопасность веб-сервиса?
42. Какие методы тестирования используются для проверки функциональности веб-сервисов?
43. Какие типы тестов могут быть применены для тестирования RESTful API, включая модульное, интеграционное и системное тестирование?
44. Какие инструменты и фреймворки доступны для автоматизации тестирования веб-сервисов?
45. Как проводится тестирование производительности веб-сервисов, и какие метрики используются для оценки их эффективности и масштабируемости?
46. Какие методы развертывания веб-сервисов используются в современных приложениях?
47. В чем состоит преимущество использования облачных платформ для развертывания веб-сервисов?
48. Какая роль у контейнеризации (например, Docker) в процессе развертывания и масштабирования веб-сервисов?
49. Что представляют собой серверы приложений, и как они используются для развертывания и управления веб-сервисами в корпоративной среде?
50. В чем разница между вертикальным и горизонтальным масштабированием, и какие преимущества и недостатки имеют эти подходы?
51. Какие факторы влияют на выбор стратегии масштабирования для конкретного веб-сервиса?
52. Какие технологии и инструменты могут помочь реализовать горизонтальное масштабирование веб-сервисов?

Вопросы разработали:

ЯЧИН Николай Сергеевич – магистр техники и технологии, старший преподаватель кафедры ПИКС

СМОРГУН Евгений Святославович – ассистент кафедры ПИКС