



Преподаватель:

Коляда

Никита Владимирович

Обратная связь:

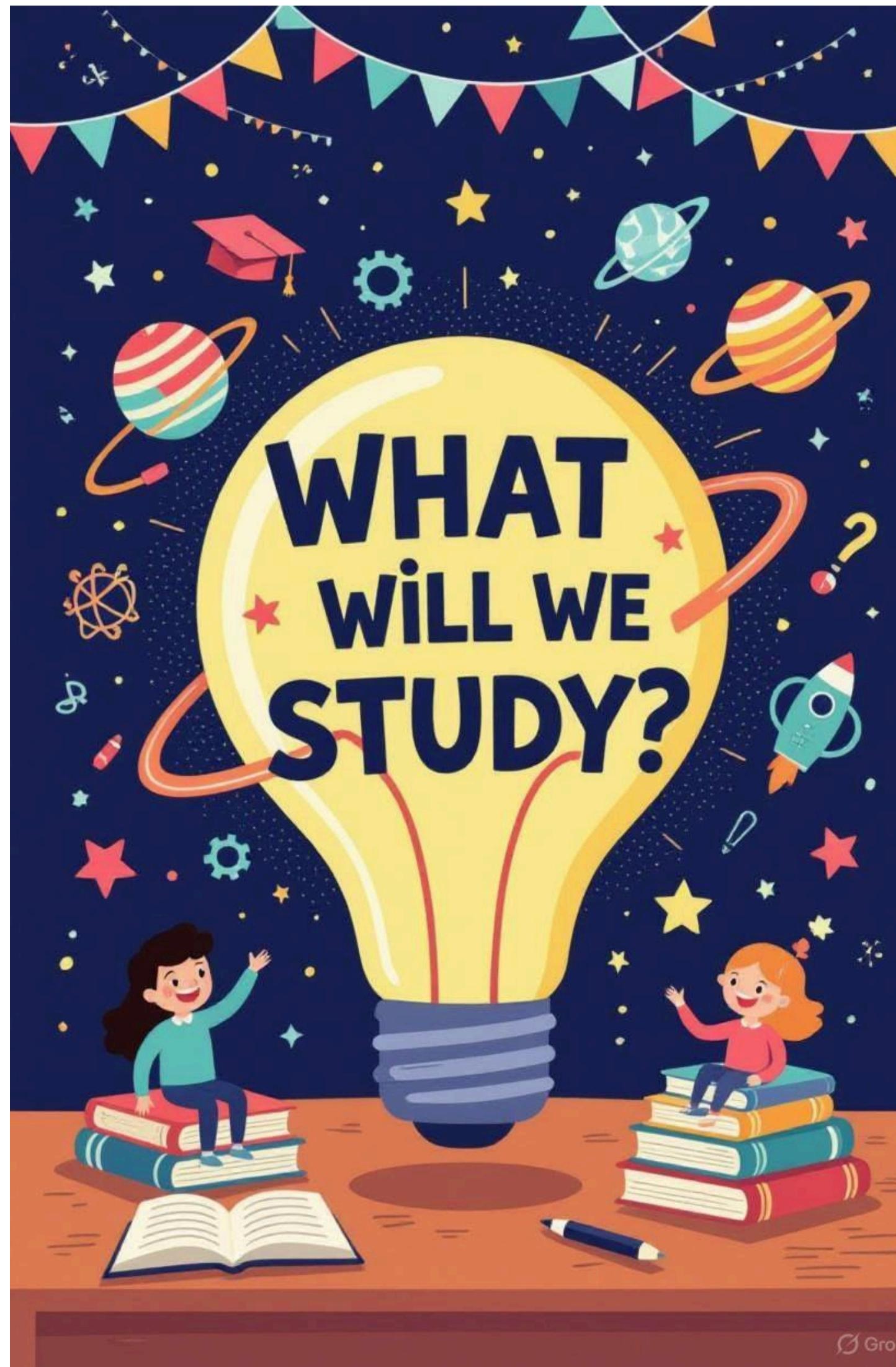
- сообщения на inStudy

2026 г.



# ВОПРОСЫ?

- ПО ТЕМАМ ДИСЦИПЛИНЫ?
- ПО ПРАКТИЧЕСКИМ/ИТОГОВЫМ РАБОТАМ?
- ПО ОРГАНИЗАЦИОННЫМ МОМЕНТАМ?

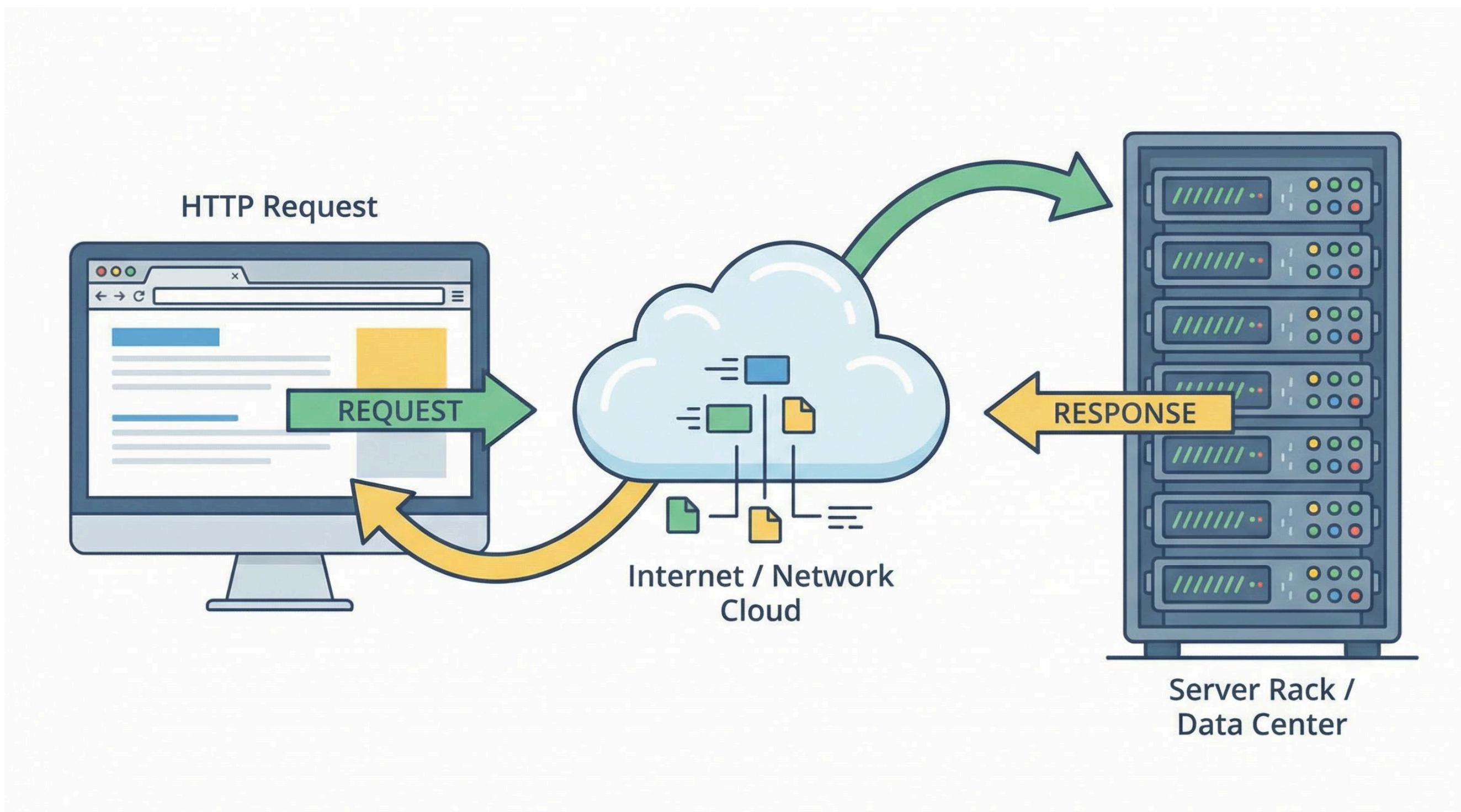


## КЛЮЧЕВЫЕ ТЕМЫ

1. Как работает интернет и веб-запросы
2. Роль браузера: что делает клиент
3. Роль сервера: зачем он нужен
4. Клиент-серверная архитектура
5. Разделение ответственности в веб-приложениях
6. Что такое backend
7. Что происходит при обработке запроса
8. Технологии для backend-разработки
9. Почему выбирают Node.js
10. Как работает Node.js (асинхронность и event loop)
11. Структура backend-приложения на Node.js
12. Взаимодействие backend и базы данных
13. Полный цикл запроса в веб-приложении

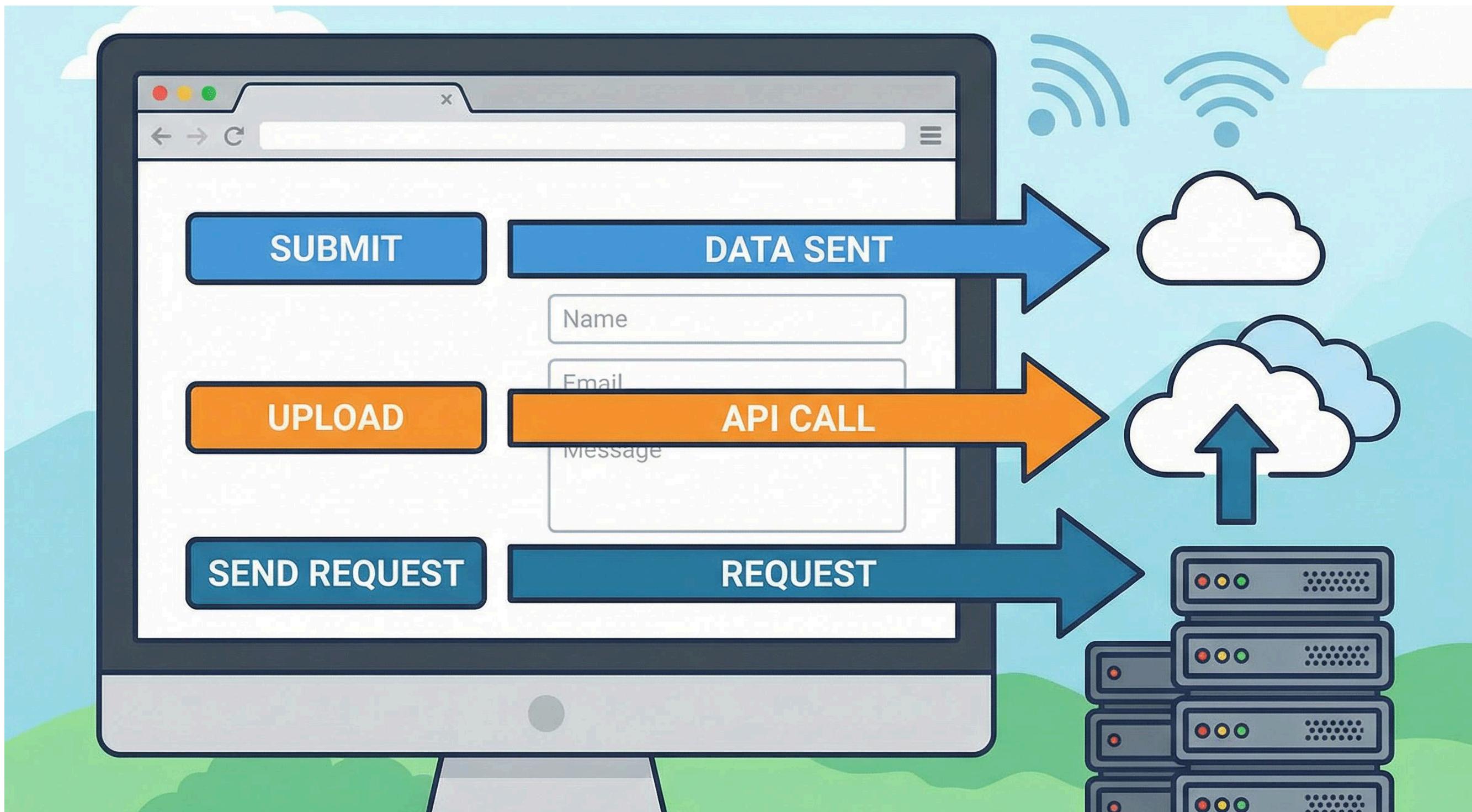
# КАК РАБОТАЕТ ИНТЕРНЕТ

ОТ БРАУЗЕРА ДО СЕРВЕРА И ОБРАТНО



1. Когда пользователь открывает сайт, браузер отправляет запрос по сети. Запрос проходит через интернет и попадает на сервер.
2. Сервер принимает запрос, обрабатывает его и отправляет ответ обратно. Ответ может содержать HTML-страницу, данные или файлы.
3. Этот обмен запросами и ответами происходит постоянно:
  - а. при загрузке страниц,
  - б. отправке форм,
  - с. нажатии кнопок.

## ЧТО ДЕЛАЕТ БРАУЗЕР КЛИЕНТСКАЯ ЧАСТЬ ПРИЛОЖЕНИЯ



1. Браузер является клиентом в системе клиент-сервер.
2. Он отвечает за:
  - отображение интерфейса
  - запуск JavaScript-кода
  - отправку запросов на сервер
3. Браузер не хранит общую базу данных приложения и не управляет пользователями напрямую.
4. Все важные операции выполняются на сервере

# КЛИЕНТ-СЕРВЕРНАЯ АРХИТЕКТУРА

## РАЗДЕЛЕНИЕ ЗАДАЧ



1. Архитектура делит приложение на две части:

клиент и сервер.

2. Клиент отвечает за интерфейс.

Сервер отвечает за данные и логику.

3. Такое разделение позволяет:

- масштабировать систему
- улучшать безопасность
- обновлять части приложения независимо

# ЧТО ТАКОЕ BACKEND

## СЕРВЕРНАЯ ЛОГИКА ПРИЛОЖЕНИЯ

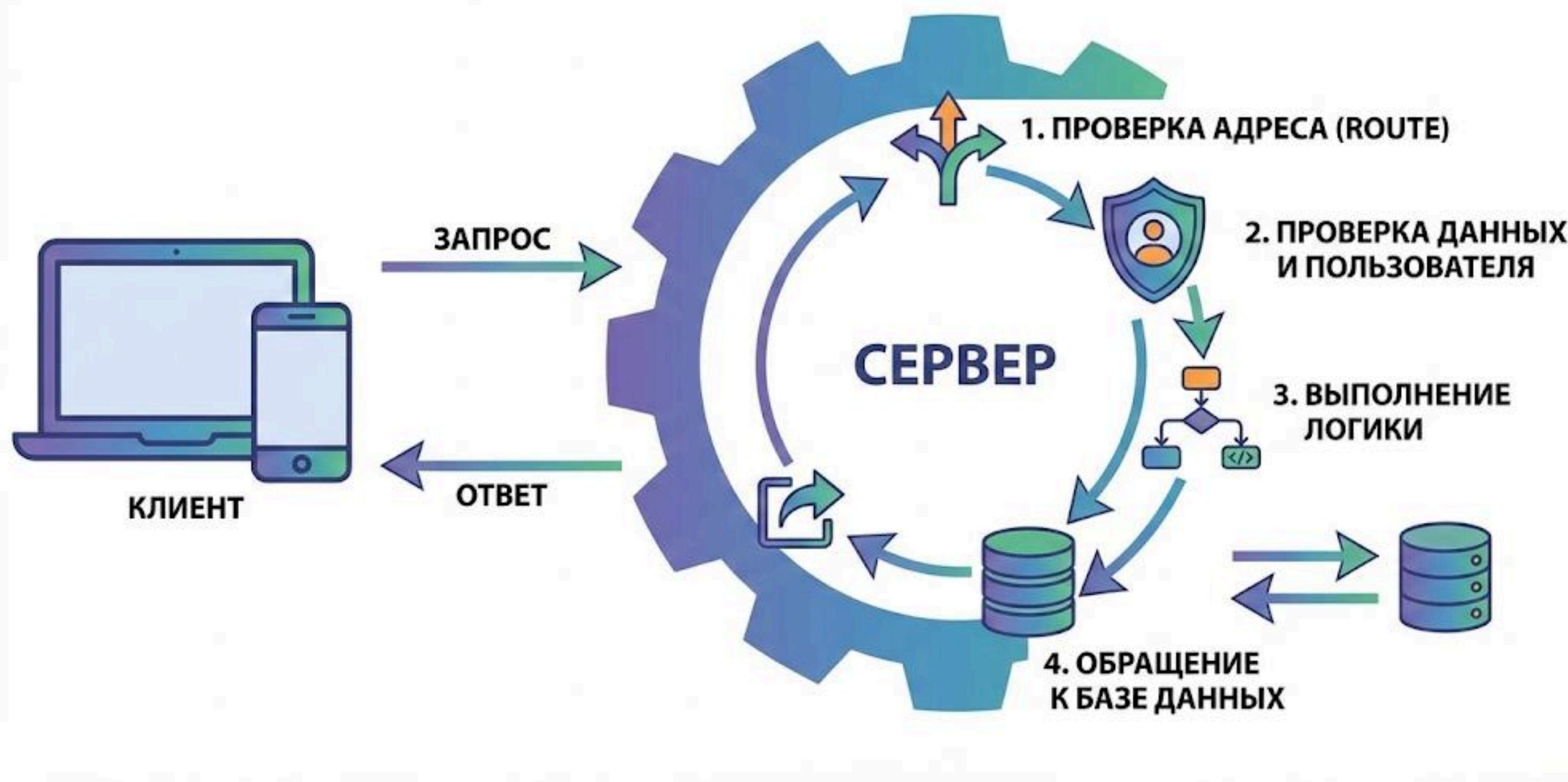


1. Backend – это программная часть, которая работает на сервере
2. Он отвечает за:
  - обработку запросов
  - авторизацию пользователей
  - работу с базами данных
  - бизнес-правила приложения
3. Пользователь не видит backend напрямую, но именно он управляет всей системой

# ЧТО ПРОИСХОДИТ ПРИ ЗАПРОСЕ

## ПУТЬ ДАННЫХ ВНУТРИ СЕРВЕРА

### ЦИКЛ ОБРАБОТКИ ЗАПРОСА НА СЕРВЕРЕ



1. Когда сервер получает запрос, он выполняет шаги:
  - а. Проверяет адрес запроса (route)
  - б. Проверяет данные и пользователя
  - с. Выполняет нужную логику
  - д. Обращается к базе данных
  - е. Возвращает ответ клиенту
2. Каждый запрос проходит этот цикл.

# ТЕХНОЛОГИИ BACKEND

НА ЧЁМ ПИШУТ СЕРВЕРЫ



1. Backend можно разрабатывать на разных языках:

- 2. – JavaScript (Node.js)
- Python (Django, Flask)
- PHP (Laravel)
- Java (Spring)
- C# (.NET)

3. Все они решают одни и те же задачи:  
принимают запросы и работают с данными

## ПОЧЕМУ NODE.JS JAVASCRIPT НА СЕРВЕРЕ

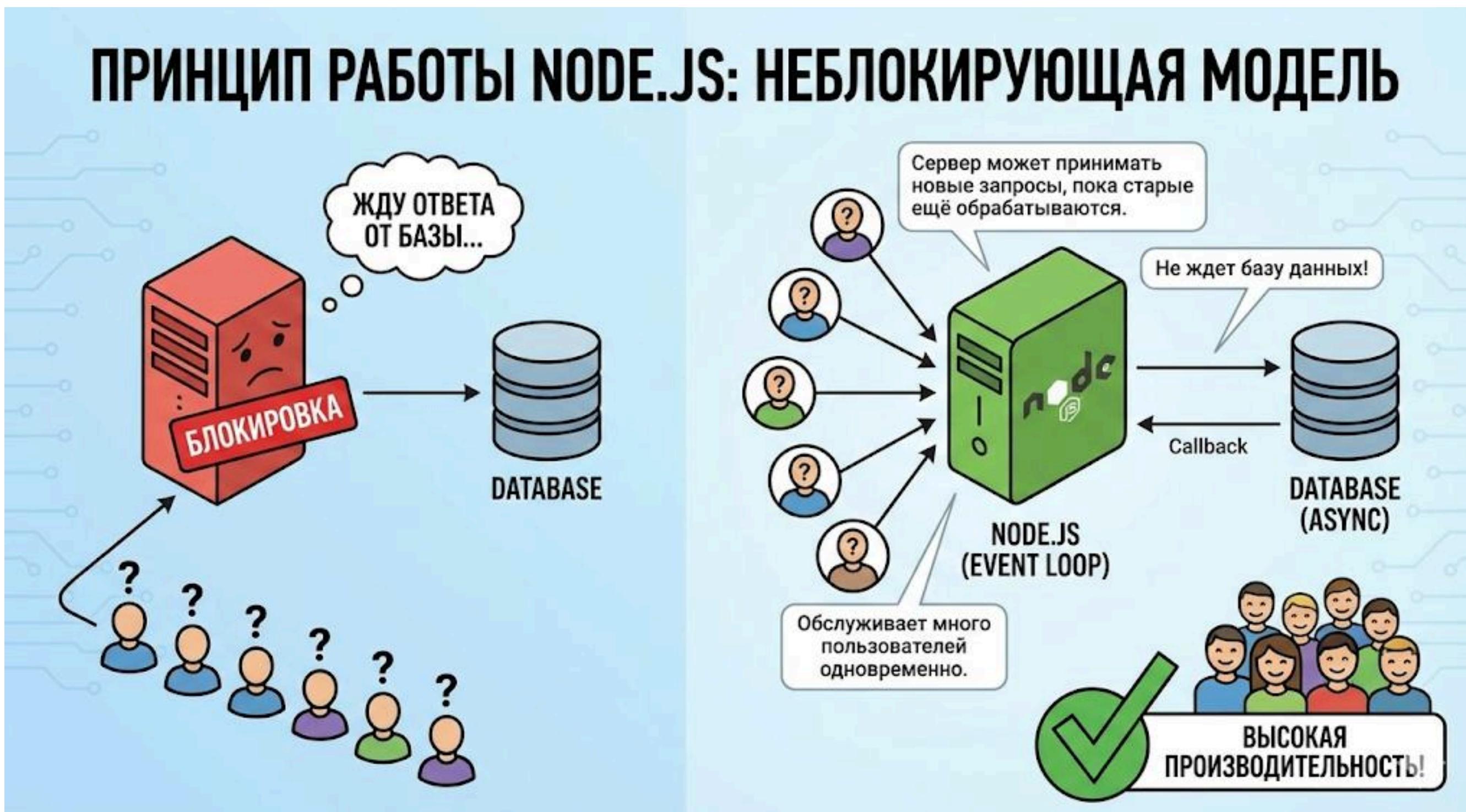


1. Node.js позволяет использовать JavaScript не только в браузере, но и на сервере.
2. Преимущества:
  - один язык для frontend и backend
  - высокая производительность
  - большое количество библиотек
  - удобен для обучения
3. Поэтому Node.js часто выбирают для старта в backend

## КАК РАБОТАЕТ NODE.JS

### АСИНХРОННАЯ ОБРАБОТКА ЗАПРОСОВ

#### ПРИНЦИП РАБОТЫ NODE.JS: НЕБЛОКИРУЮЩАЯ МОДЕЛЬ

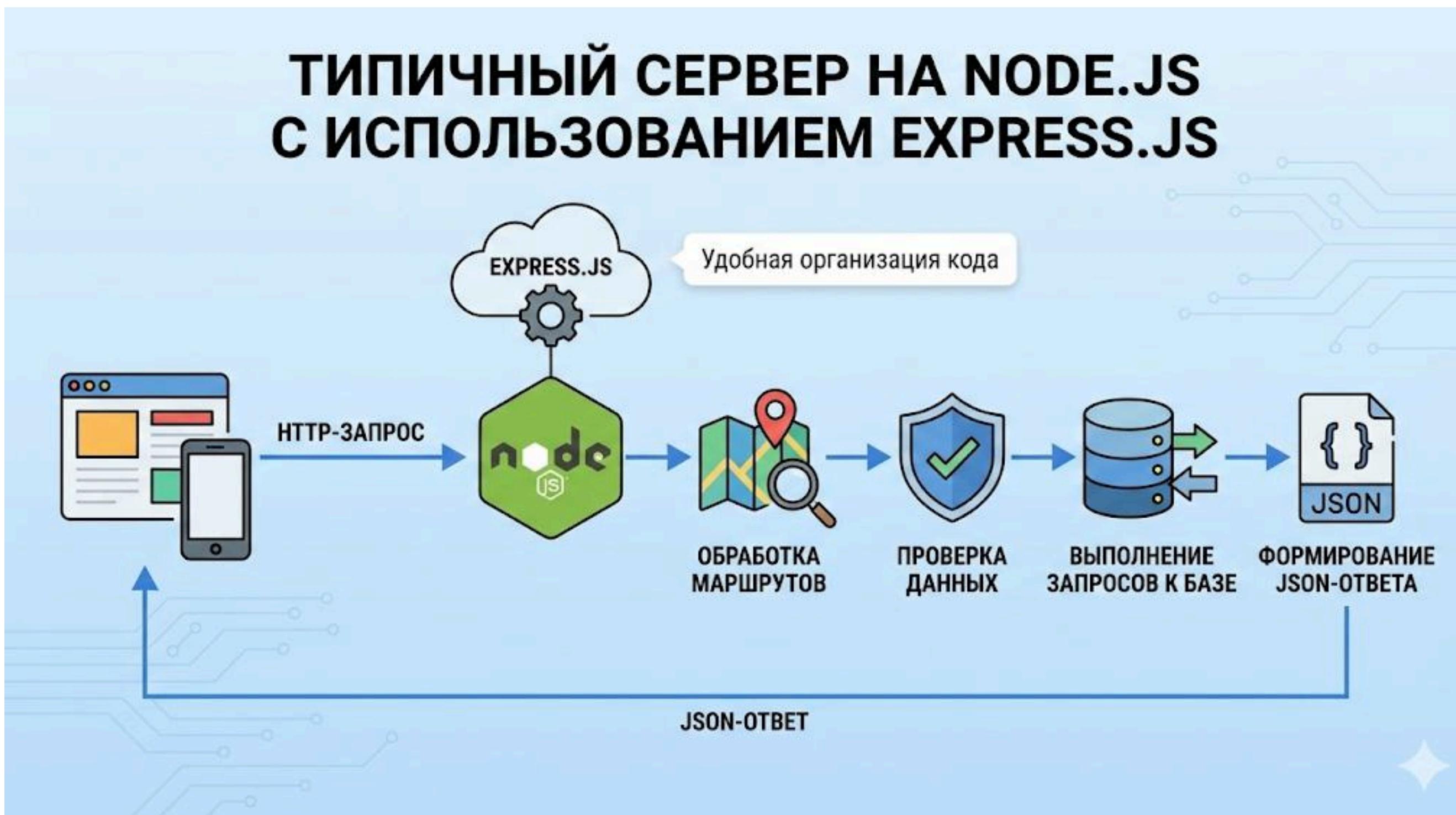


1. Node.js работает по неблокирующему принципу.
2. Это означает: сервер может принимать новые запросы, пока старые ещё обрабатываются.
3. Вместо ожидания ответа от базы данных Node.js продолжает обслуживать других клиентов.
4. Это позволяет обслуживать много пользователей одновременно.

# СЕРВЕР НА NODE.JS

ЧТО ОБЫЧНО ДЕЛАЕТ BACKEND

## ТИПИЧНЫЙ СЕРВЕР НА NODE.JS С ИСПОЛЬЗОВАНИЕМ EXPRESS.JS



1. Типичный сервер на Node.js:
  - принимает HTTP-запросы
  - обрабатывает маршруты
  - проверяет данные
  - выполняет запросы к базе
  - возвращает JSON-ответы
2. – часто используется фреймворк Express.js для удобной организации кода

# BACKEND И БАЗА ДАННЫХ

СВЯЗЬ СЕРВЕРА И MYSQL



1. Backend не хранит данные сам.

Он подключается к базе данных.

2. Через SQL-запросы сервер:

- получает данные
- сохраняет изменения
- удаляет записи

3. Пользователь никогда не подключается к БД напрямую — только через сервер

## ПОЛНЫЙ ЦИКЛ РАБОТЫ ОТ КЛИКА ДО БАЗЫ ДАННЫХ



1. Полный путь запроса:
2. Пользователь нажал кнопку → браузер отправил запрос → сервер обработал → база вернула данные → сервер отправил ответ → браузер обновил интерфейс
3. Это основа работы всех современных веб-приложений



# ВОПРОСЫ?

- ПО ТЕМАМ ДИСЦИПЛИНЫ?
- ПО ПРАКТИЧЕСКИМ/ИТОГОВЫМ РАБОТАМ?
- ПО ОРГАНИЗАЦИОННЫМ МОМЕНТАМ?