

# переизобретая BFF

как Яндекс Маркет  
переделал архитектуру  
клиентского сервиса

**Егор Быховцев**

Ведущий разработчик интерфейсов



ГОРОДСКИЕ  
СЕРВИСЫ



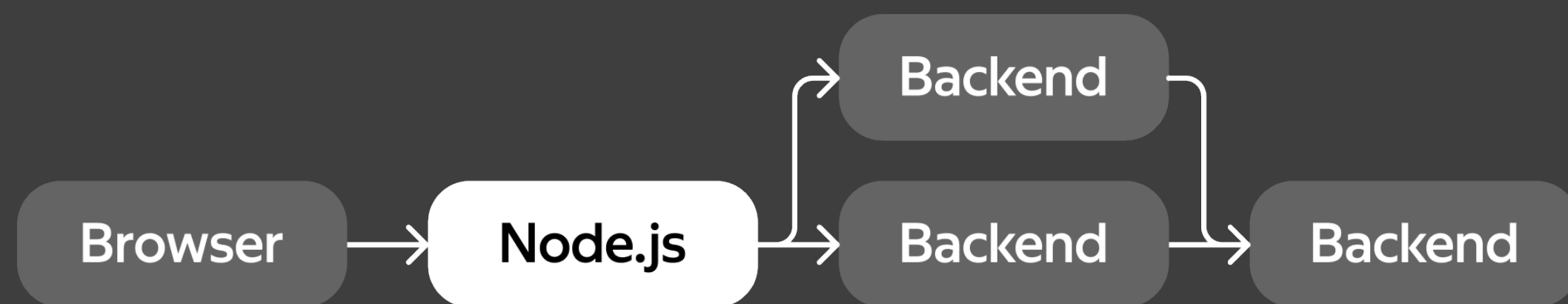
# привет!

- 01 Окончил МехМат НГУ
- 02 9 лет в Яндексе
- 03 Расту вместе с Маркетом
- 04 Занимаюсь архитектурой микросервисов

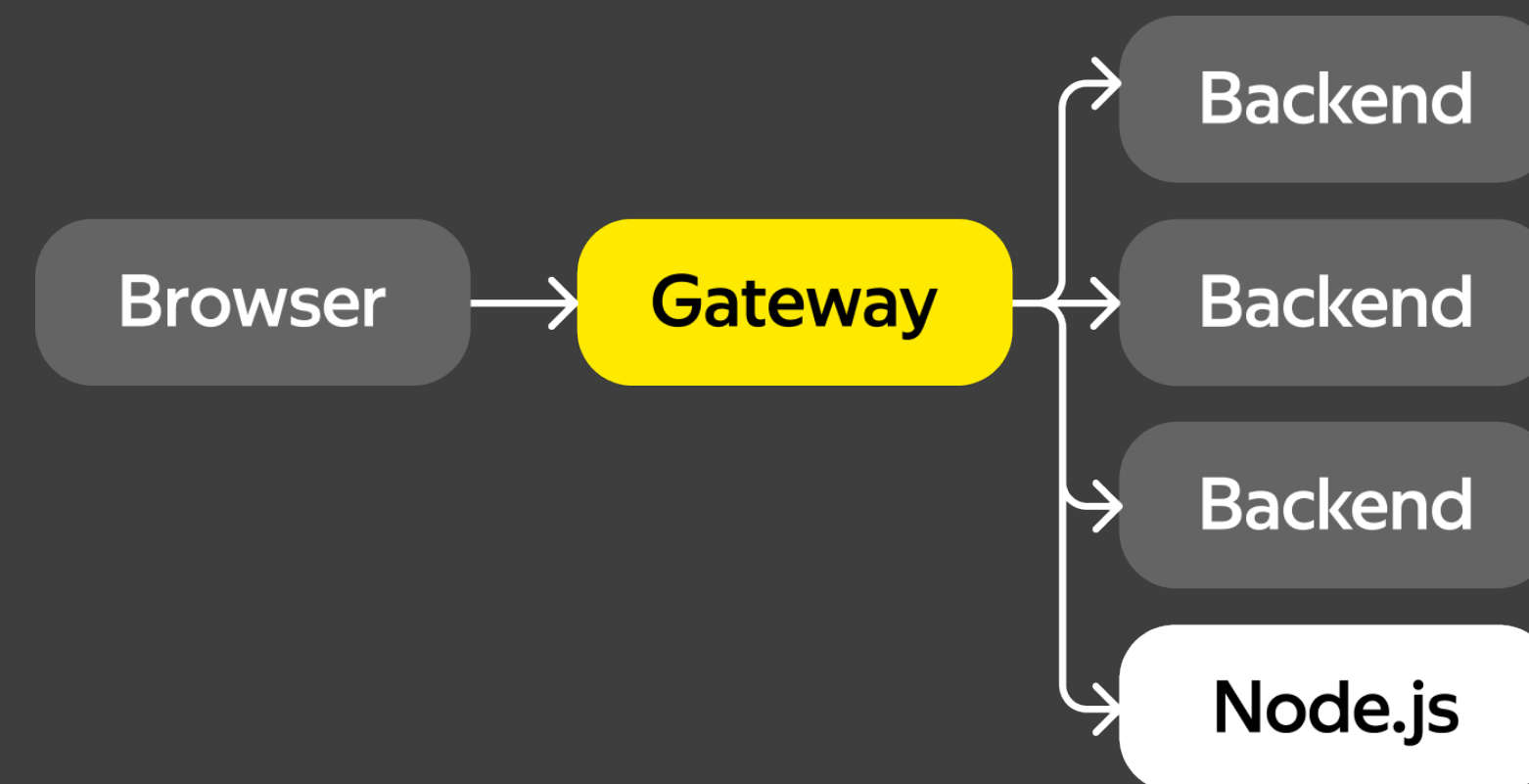


# tldr;

Было  
неправильно и плохо



Стало  
правильно и хорошо



# О ЧЁМ ПОГОВОРИМ

Как мы готовили Маркет

01

Что нам не нравилось

02

Дизайн новой архитектуры

03

Что получилось?

04

Заключение

05



# КАК МЫ ГОТОВИЛИ?





# Яндекс Маркет

Показываем контент,  
Есть интерактивность

Миллионный DAU

Много поверхностей  
показа: App, Web, Go

Требования к SEO



Каталог

Найти товары

Призы

Колесо призов



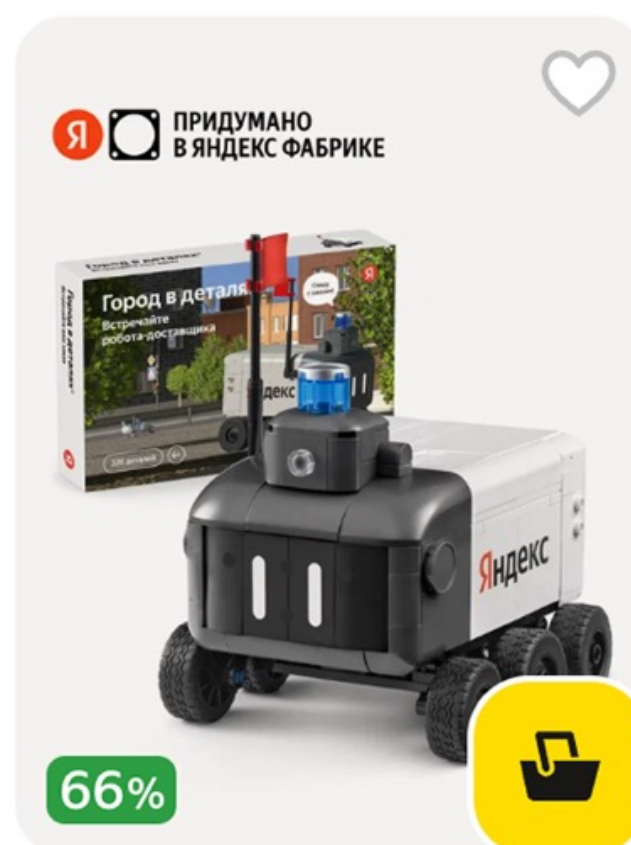
Заказы

Для вас

Ниже рынка



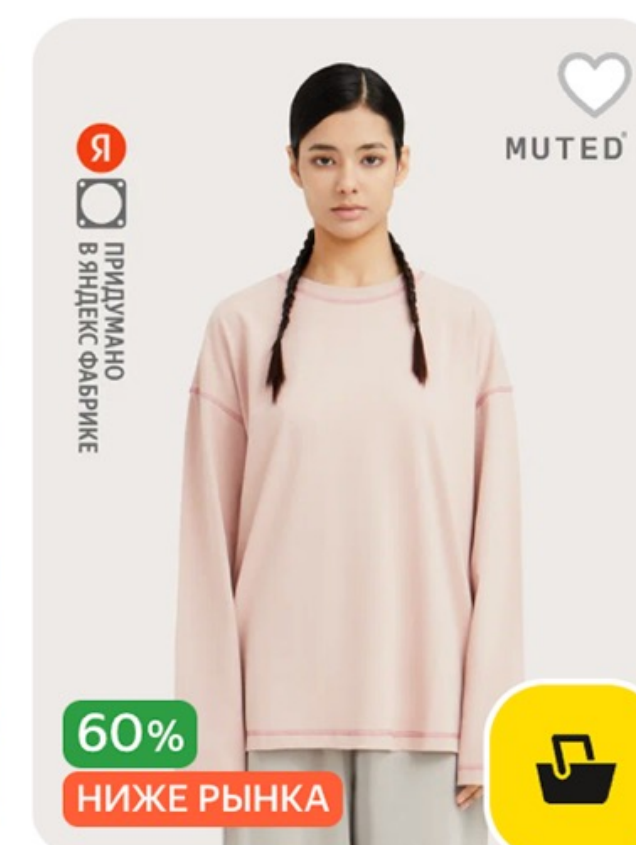
**4 473 руб.** Альфа  
4×1119 руб. в сплит  
Фен-щетка с ионизацией  
Tuvio HBI1412, 7 насадок,...  
★ 4.8 70172 оценки  
● 13 ноя, ПВЗ · Курьер



**1 516 руб.** Альфа  
4×379 руб. в сплит  
Конструктор «Город в  
деталях. Встречайте...  
★ 4.7 2178 оценок  
● Завтра, ПВЗ · Курьер



**865 руб.** Альфа  
4×217 руб. в сплит  
Ёлочная игрушка лошадка  
в подарочной упаковке...  
★ 4.9 152 оценки  
● Завтра, ПВЗ · Курьер



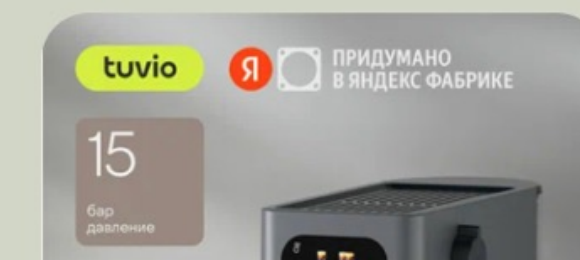
**1 188 руб.** Альфа  
4×297 руб. в сплит  
Лонгслив MUTED  
CONTRAST  
● 13 ноя, ПВЗ · Курьер

Рекомендуем вам

Горячие напитки



**6 900 руб.** Альфа



**6 900 руб.** Альфа



# Яндекс Маркет

Миллионы строк кода

Отрисовка через  
Node.js и Spring

Прогрессивный SSR

Десятки бэкендов



Каталог

Найти товары

Призы

Колесо призов



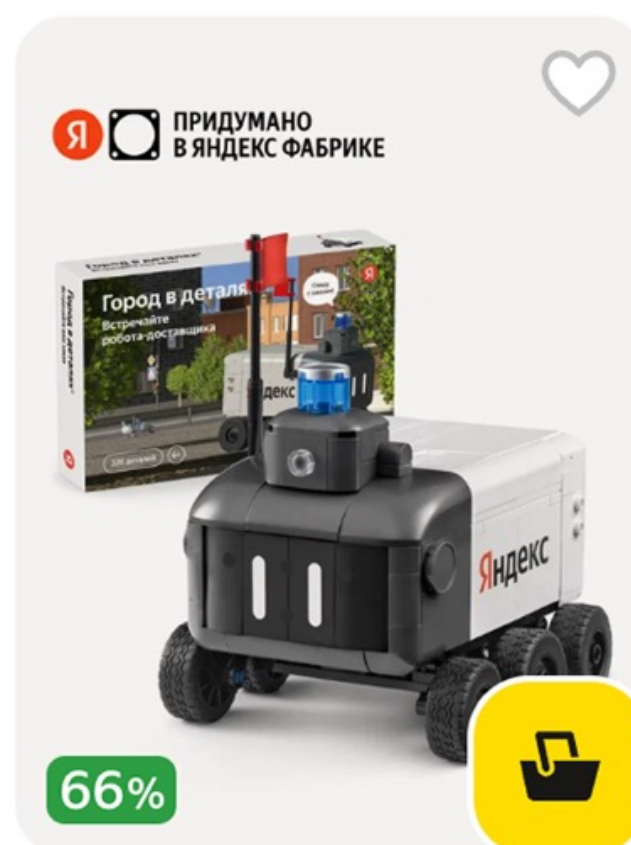
Заказы

Для вас

Ниже рынка



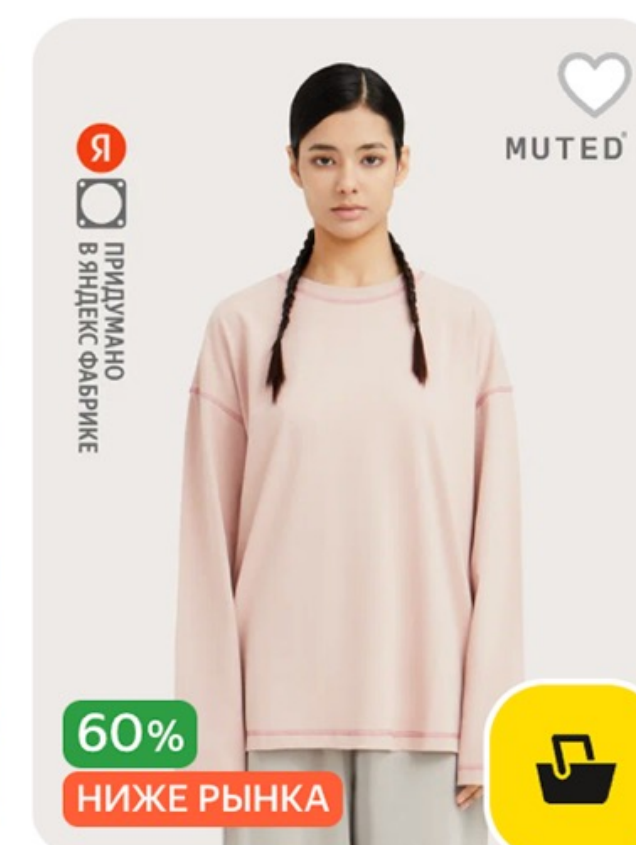
**4 473 ₽** Альфа  
4×1 119 ₽ в сплит  
Фен-щетка с ионизацией  
Tuvio HBI1412, 7 насадок,...  
★ 4.8 70172 оценки  
● 13 ноя, ПВЗ · Курьер



**1 516 ₽** Альфа  
4×379 ₽ в сплит  
Конструктор «Город в  
деталях. Встречайте...  
★ 4.7 2178 оценок  
● Завтра, ПВЗ · Курьер



**865 ₽** Альфа  
4×217 ₽ в сплит  
Ёлочная игрушка лошадка  
в подарочной упаковке...  
★ 4.9 152 оценки  
● Завтра, ПВЗ · Курьер

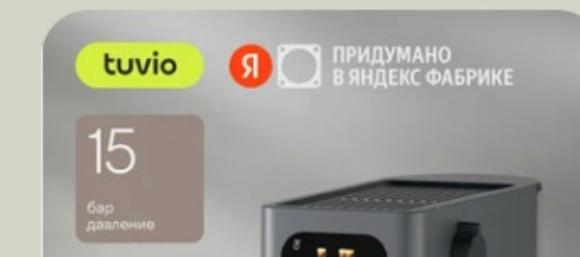
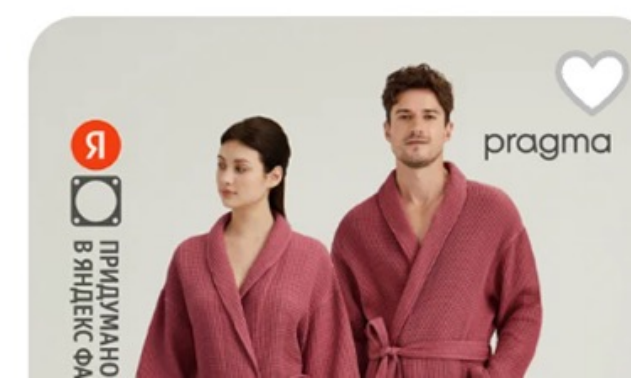
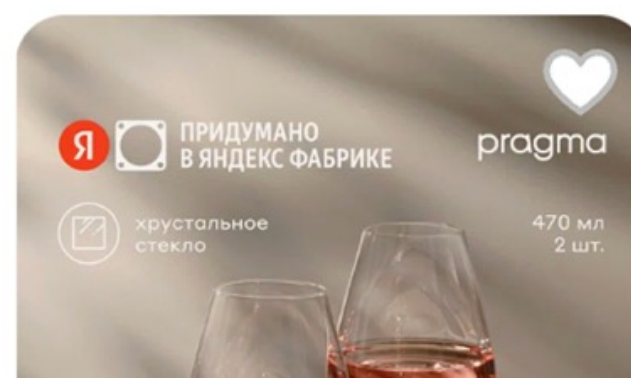
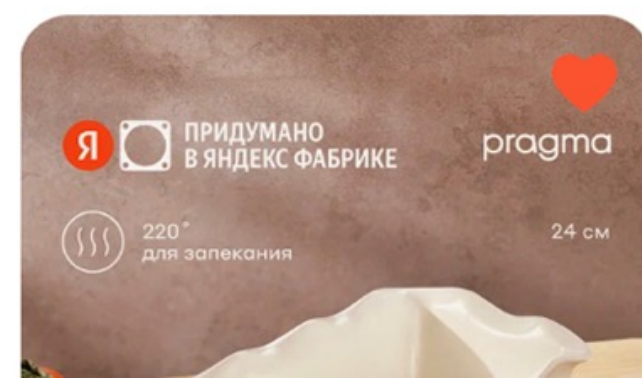


**1 188 ₽** Альфа  
4×297 ₽ в сплит  
Лонгслив MUTED  
CONTRAST  
● 13 ноя, ПВЗ · Курьер

Рекомендуем вам  
Горячие напитки



**6 900 ₽** Альфа





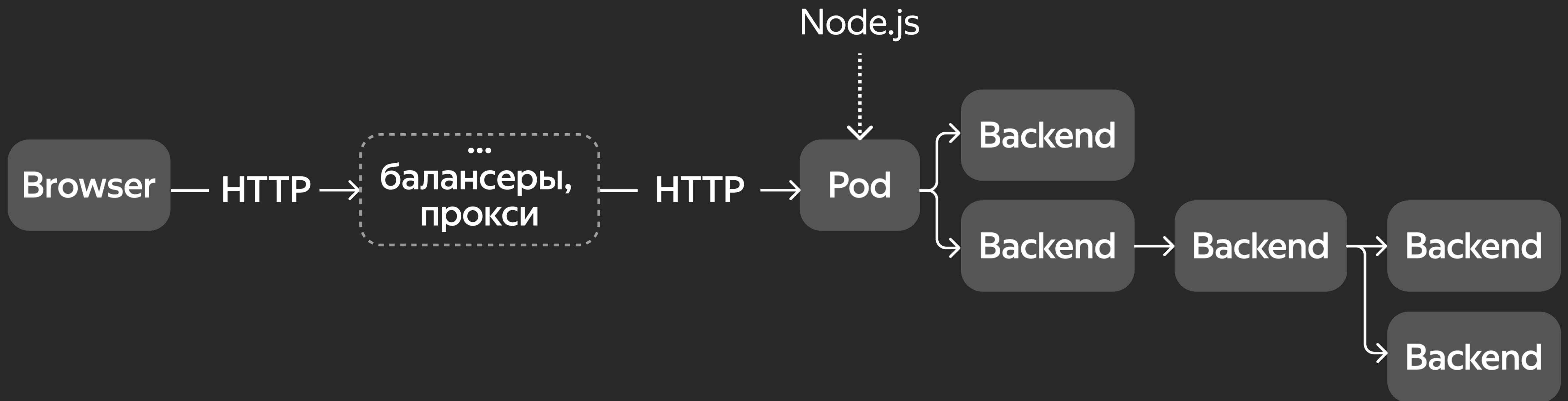
# прогрессивная отдача

- 01 Страница отдаётся блок за блоком. Кусочек HTML за кусочком
- 02 Требуется стриминг ответа  
Transfer-Encoding: chunked

slow 4g,  
6x slowed video

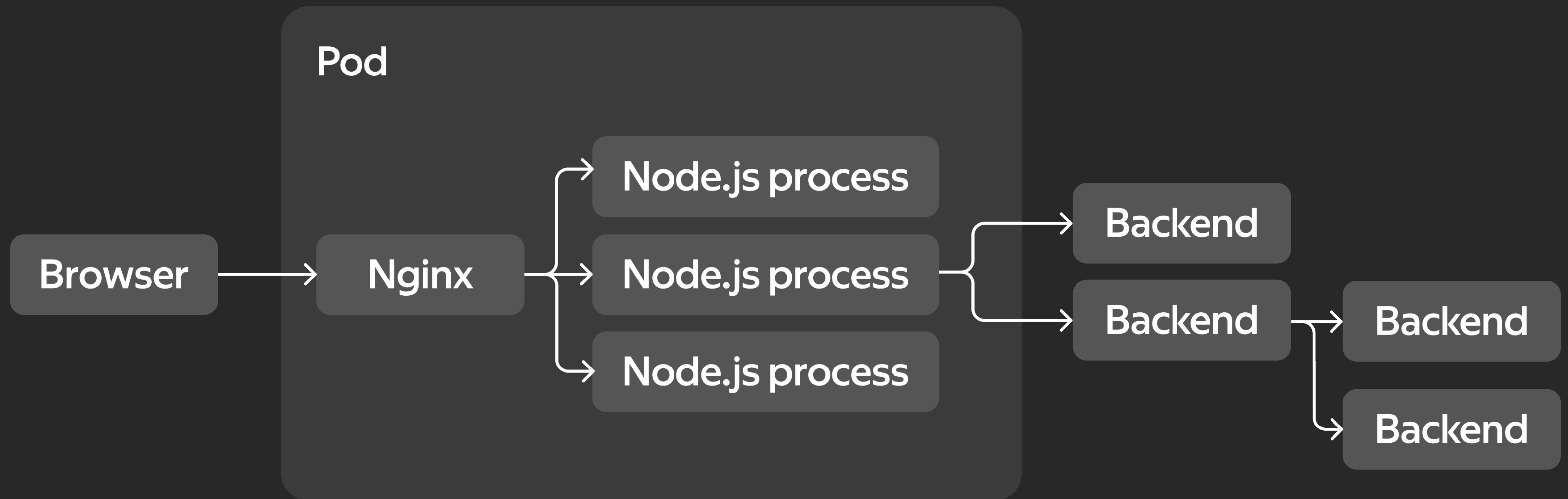
# как это работает

## ВЗГЛЯД СВЕРХУ



как это работает

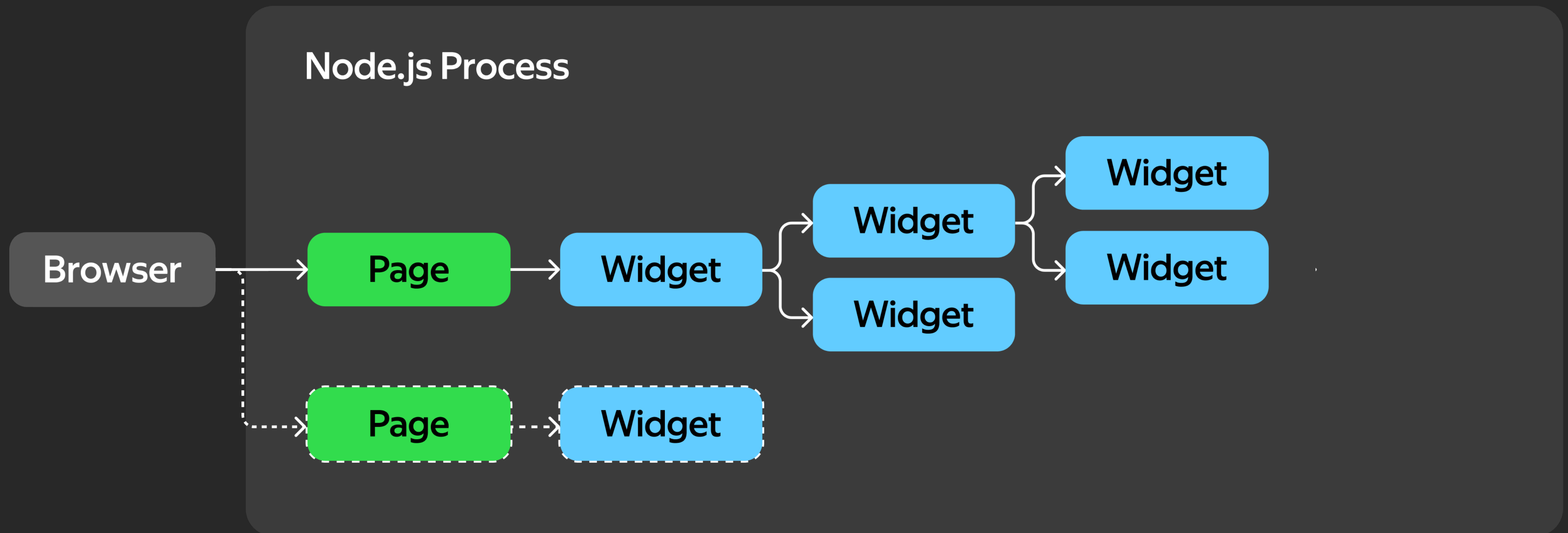
# ЧТО ПРОИСХОДИТ ВНУТРИ ПОДА



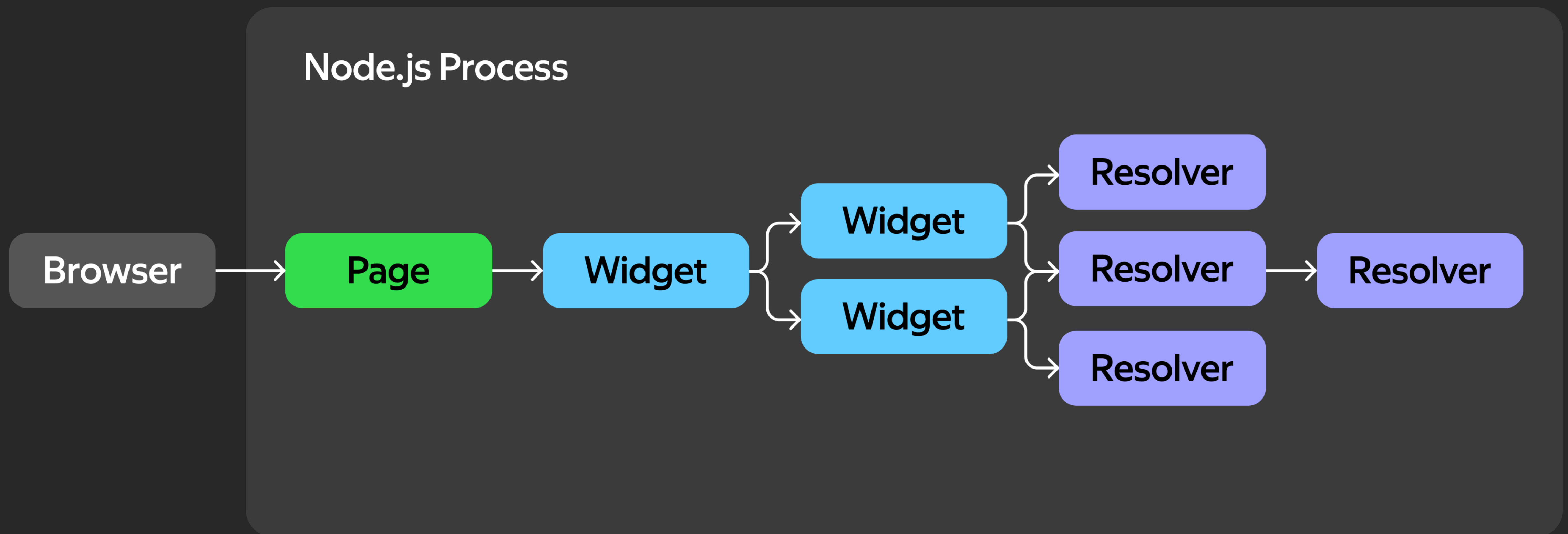


как это работает

# что происходит внутри процесса



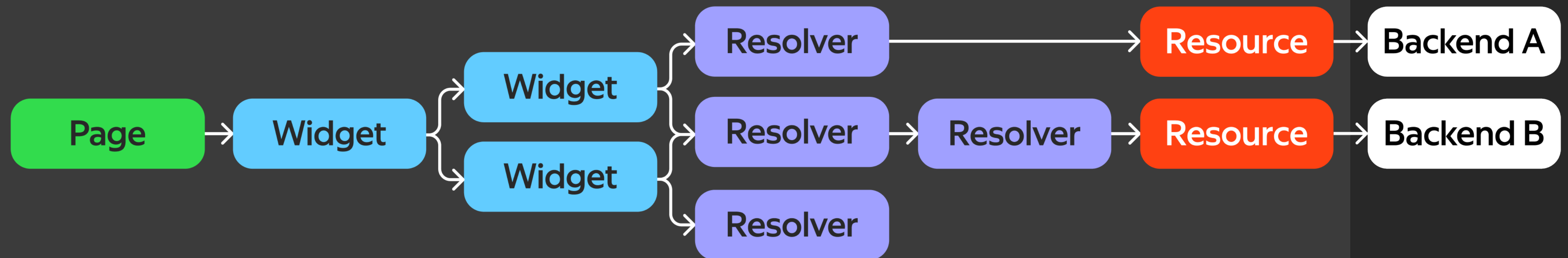
# что происходит внутри процесса ГОТОВИМ данные



# что происходит внутри процесса

## ХОДИМ К СОСЕДЯМ

Node.js Process





Widget

Resolver

Resource

```
Widget {  
  controller() {  
    return resolveOrders()  
  }  
  view(orders) {  
    return (  
      <ul>  
        {orders.map(order => (  
          <li>  
            <Order order={order}>  
          </li>  
        )}  
      <ul>  
    );  
  }  
}
```

```
resolveOrders() {  
  // Получаем заказы пользователя  
  const orders = ordersResource(userId);  
  // Получаем адрес по геопозиции  
  const addresses = geoResource(  
    orders.map(getCoordinate)  
  )  
  
  // Дозаполняем заказы адресами  
  return orders.map((i, order) => ({  
    ...order  
    address: addresses[i]  
  }));  
}
```

```
ordersResource() {  
  url: backend.yandex.net  
  protocol: http  
  port: 80  
  retry: 1  
  prepare: (userId) => {...}  
  onError: () => {...}  
}
```

## Виджет

Связывание данных  
и шаблона вёрстки

## Резолвер

Шейпинг данных  
и бизнес-логика

## Ресурс

Получение данных  
от бэкенда  
Много инфры тут

# а что не так?





# Event Loop Lag

01

ELL — это задержка в обработке событий Node.js

02

ELL — это когда занята js-ом и не делает ничего другого

03

Основной триггер — долгие синхронные операции.  
Например, `React.render`

04

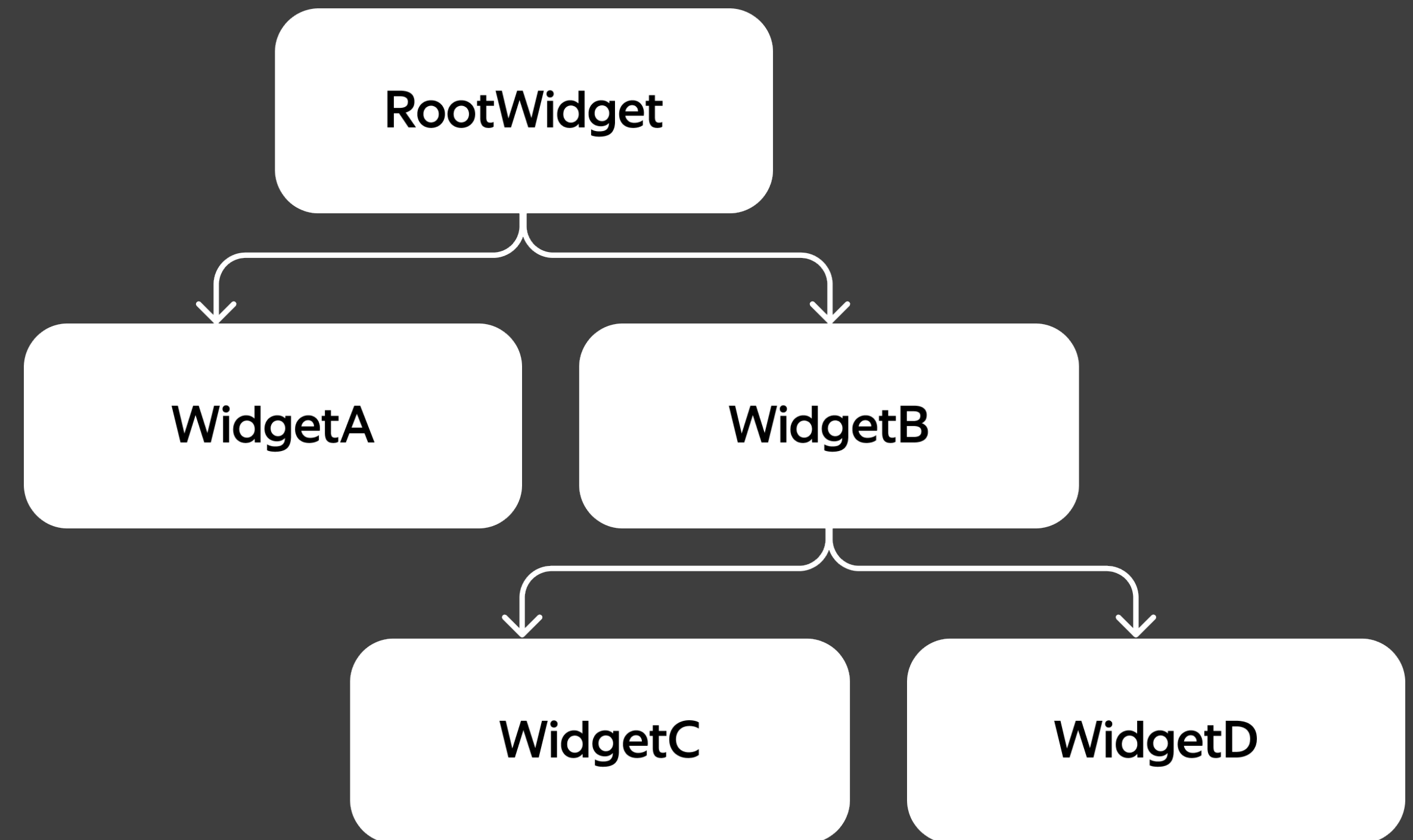
Корневая причина — однопоточность языка

# состав виджета

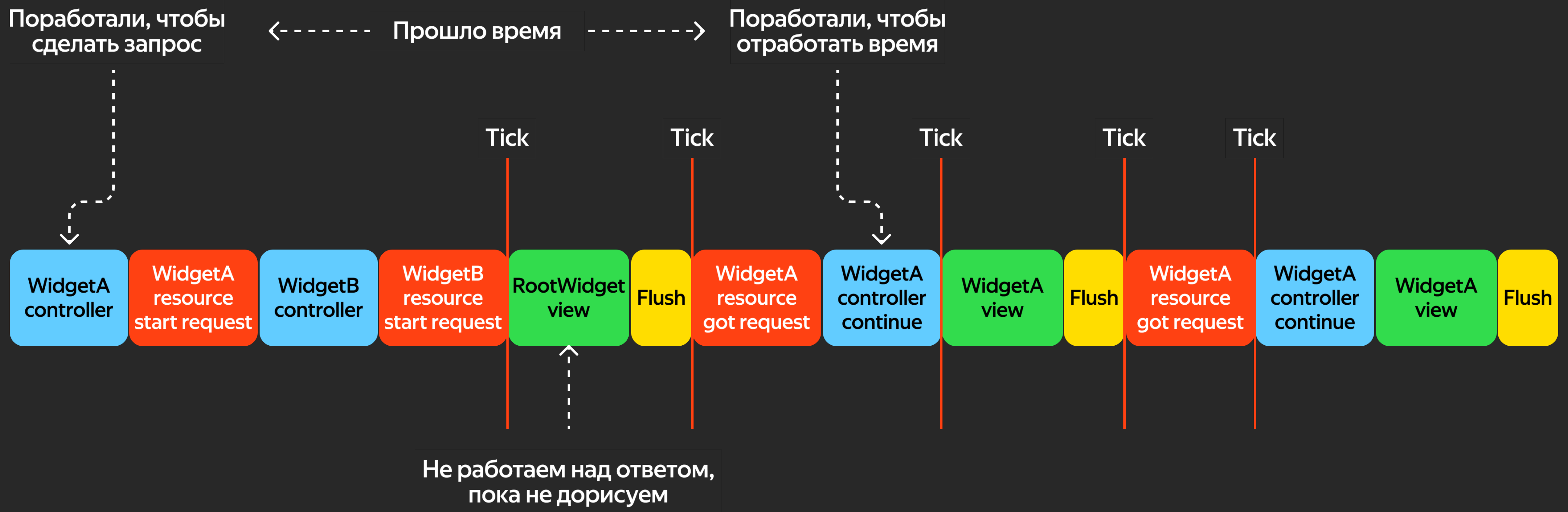
Виджет связывает

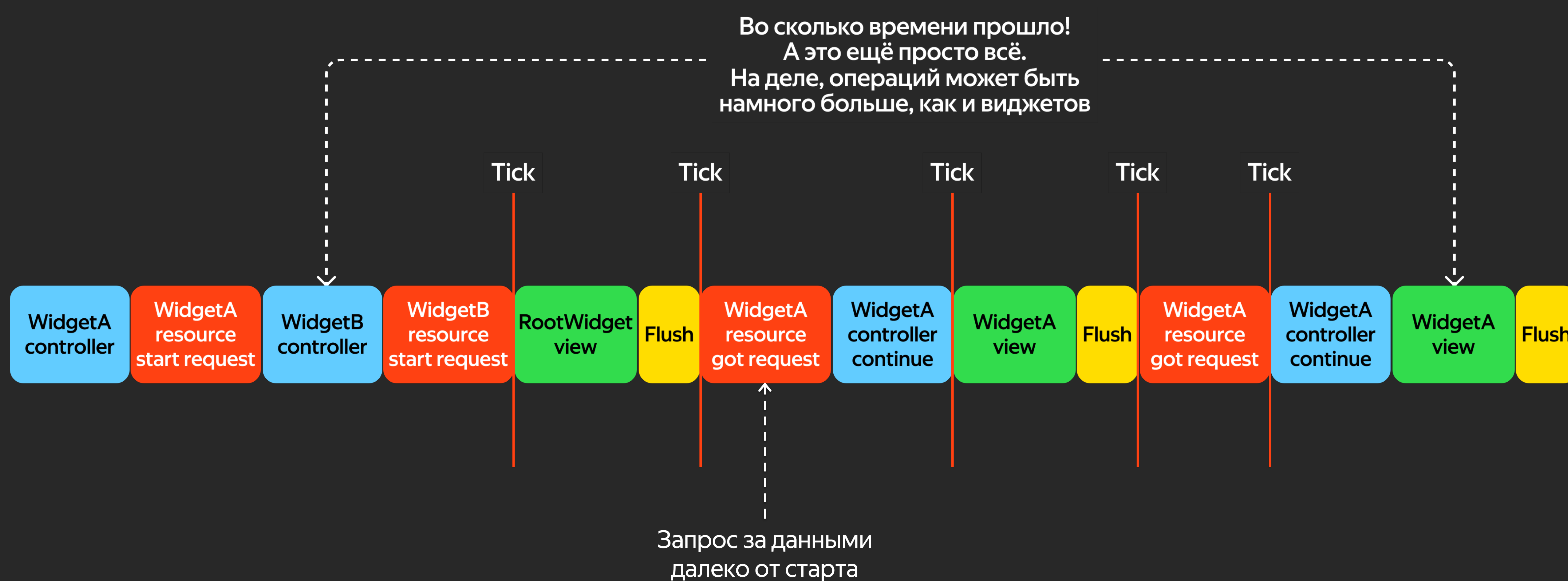
- получение данных (controller)
- отображение (view)

Виджеты вызывают контроллеры и вьюшки друг друга











# способы решения

01

Внедрить очередь приоритетов

02

Распареллелить отрисовку на честные потоки

03

Оптимизировать отрисовку компонентов

## что хорошо

- 01 Динамический состав страницы
- 02 Прогрессивная отдача
- 03 Шустро работает
- 04 Деление ответственности

## что плохо

- 01 Проблемы с TTFB
- 02 Проблемы с ELL
- 03 Очень много логики внутри Node.js
- 04 Налог на инфраструктуру
- 05 Мобильные приложения делают ту же работу

# Новая архитектура

решаем все проблемы за один подход



# требования

ЧТО МЫ ХОТИМ ПОЛУЧИТЬ ОТ НОВОЙ архИ

## reuse

Код работы с данными  
должен переиспользоваться  
между Web и App

## zero i/o

Node.js сервис перестаёт  
ходить по сети —  
нам не нравится налог  
на инфраструктуру

## TTFB

Должны ускориться  
в ключевых сценариях SEO

## balance

Не должны переписать  
всё в процессе

## progressive

Мы не должны потерять  
прогрессивность отдачи

## advert

Место для вашей рекламы

# дизайн архитектуры

01

Делаем новый компонент перед BFF

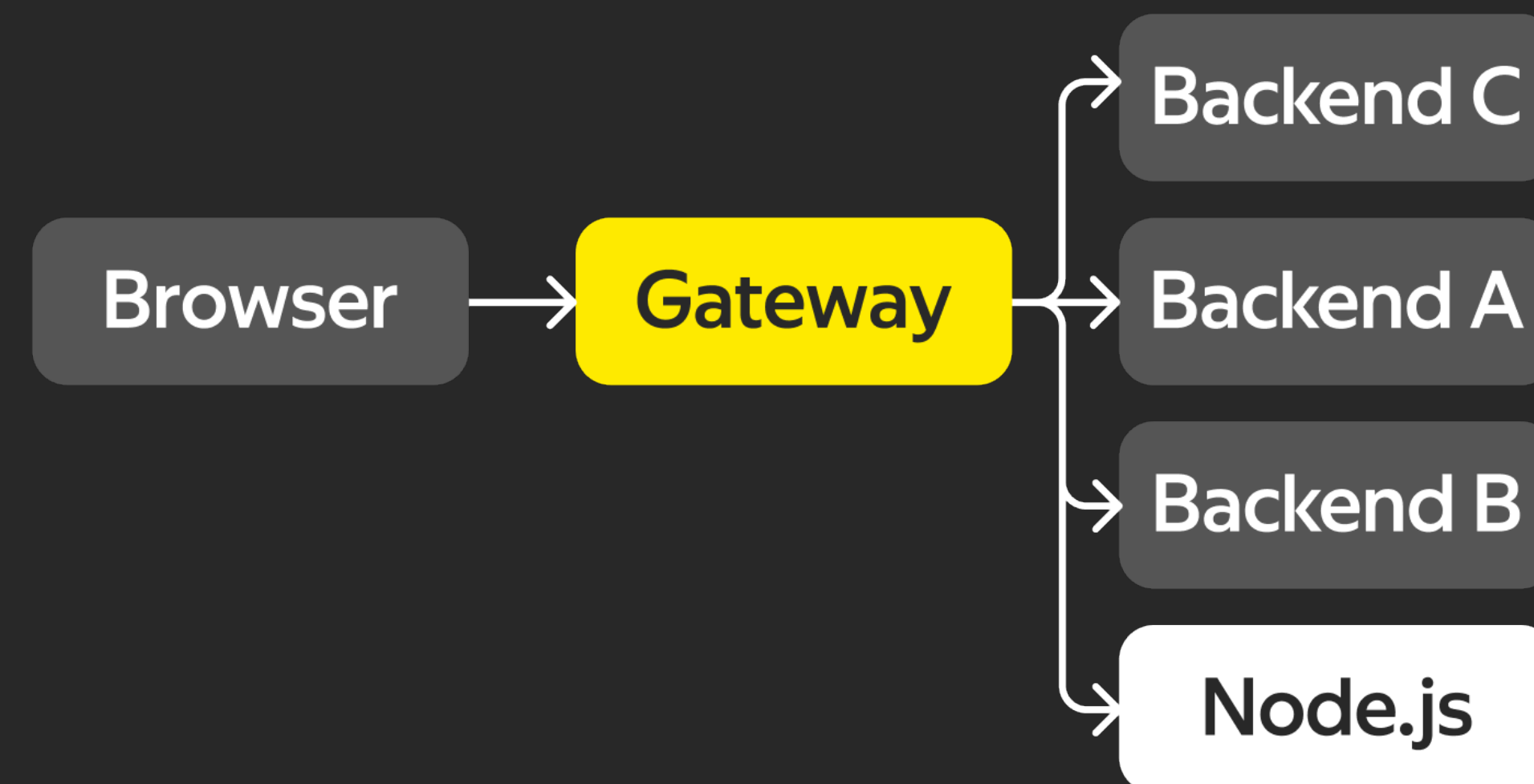
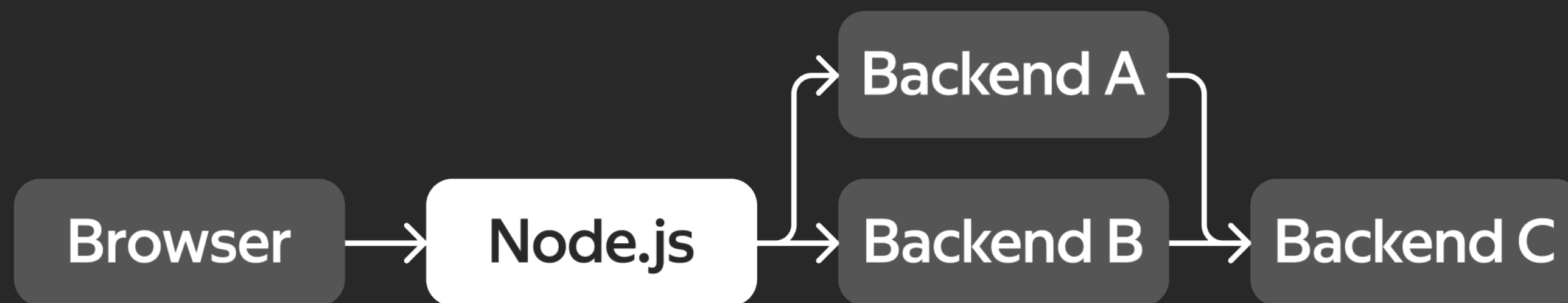
02

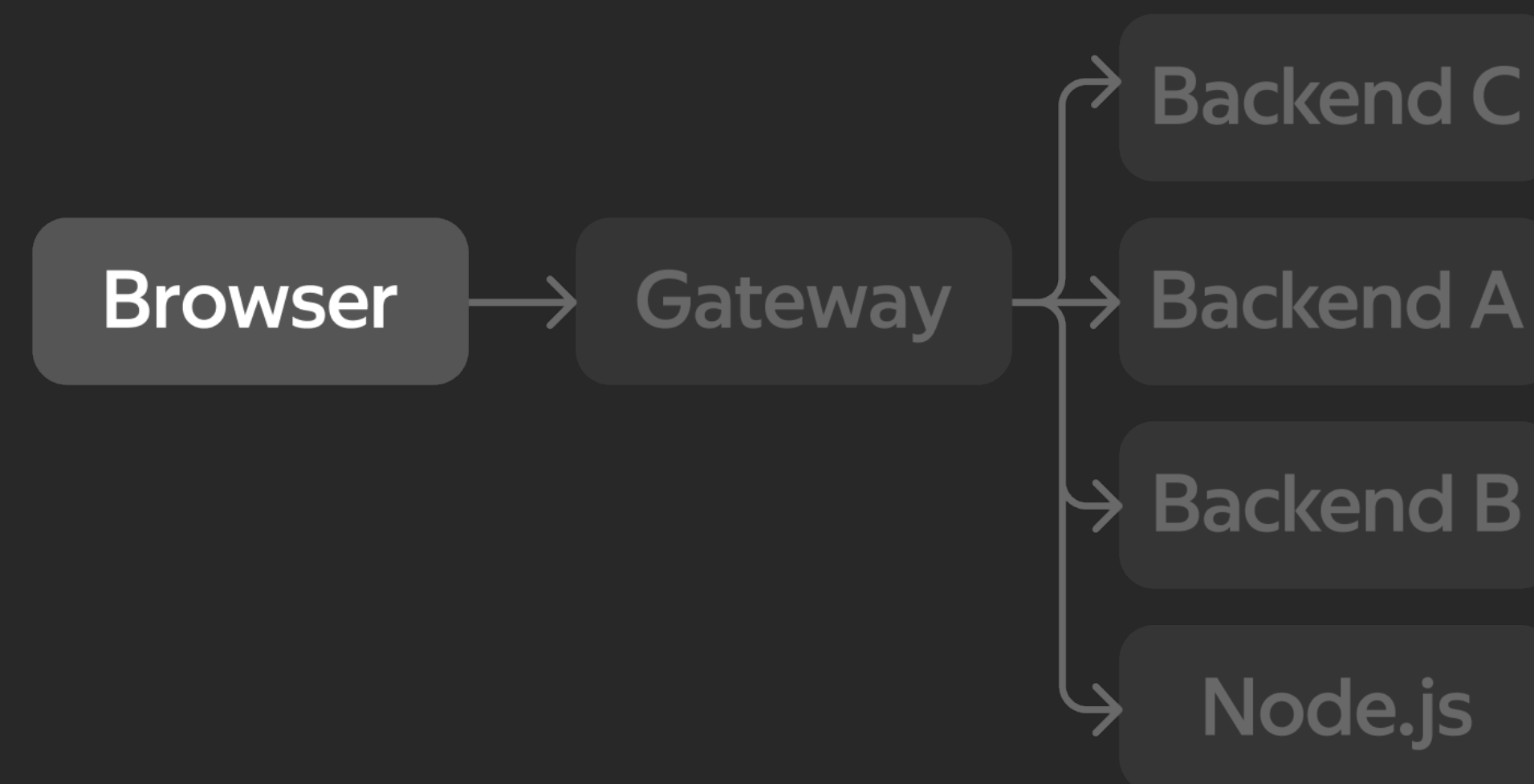
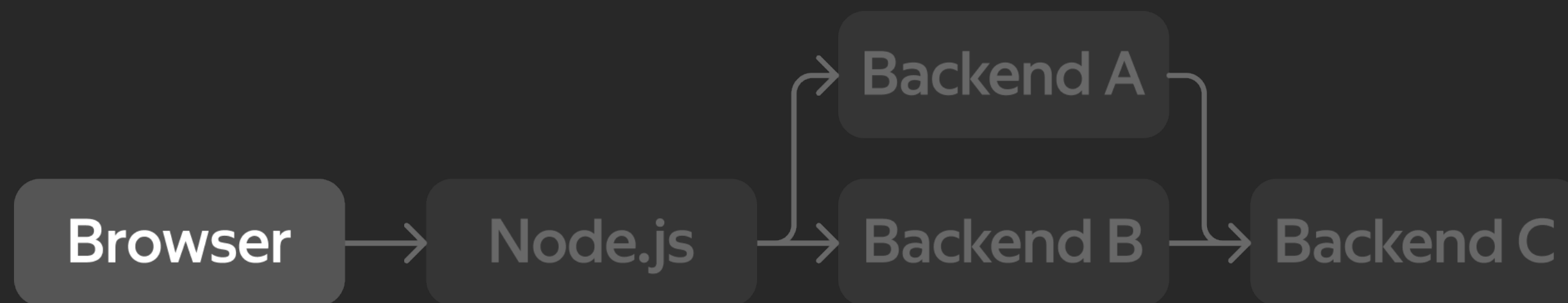
Компонент оркестрирует запросы

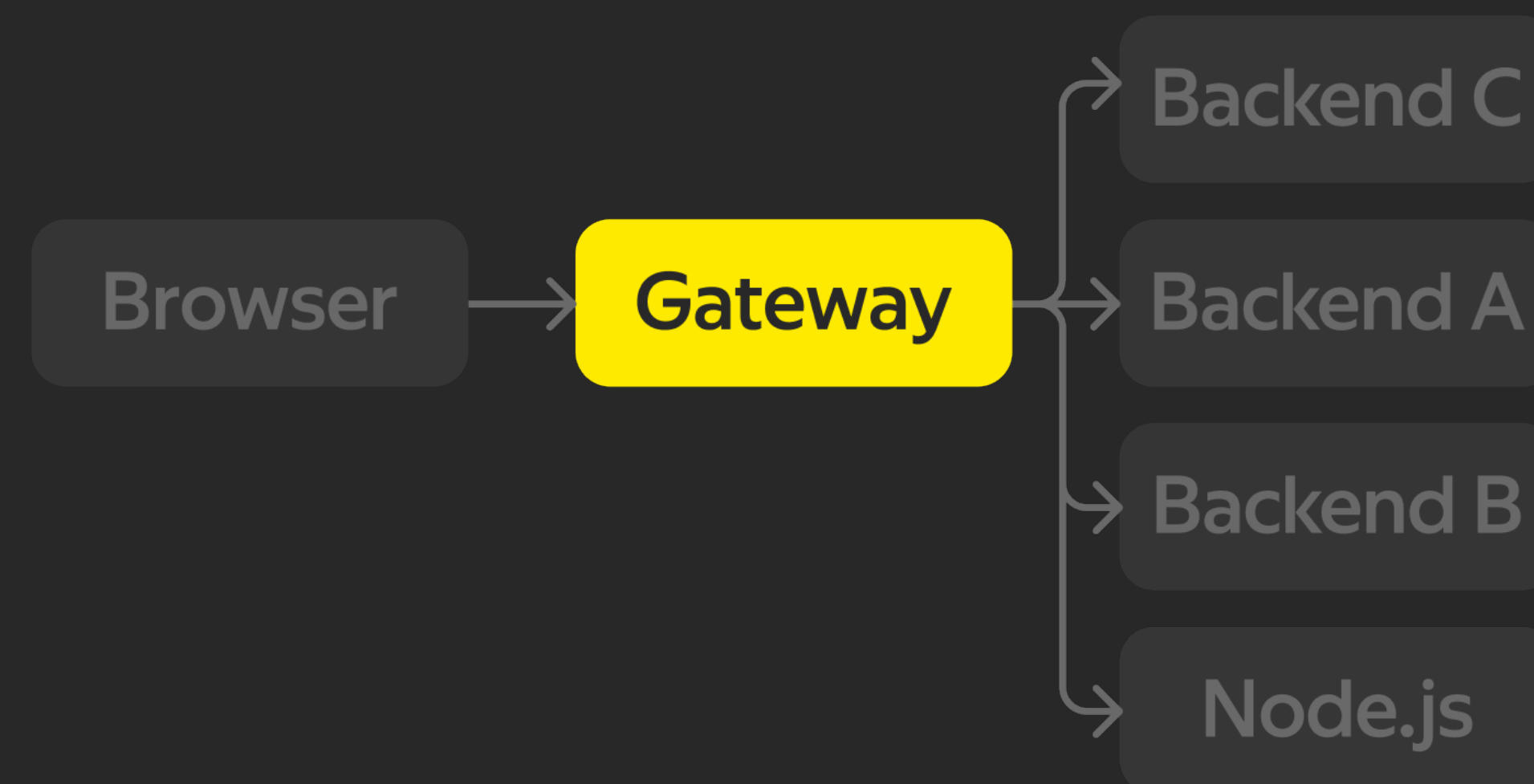
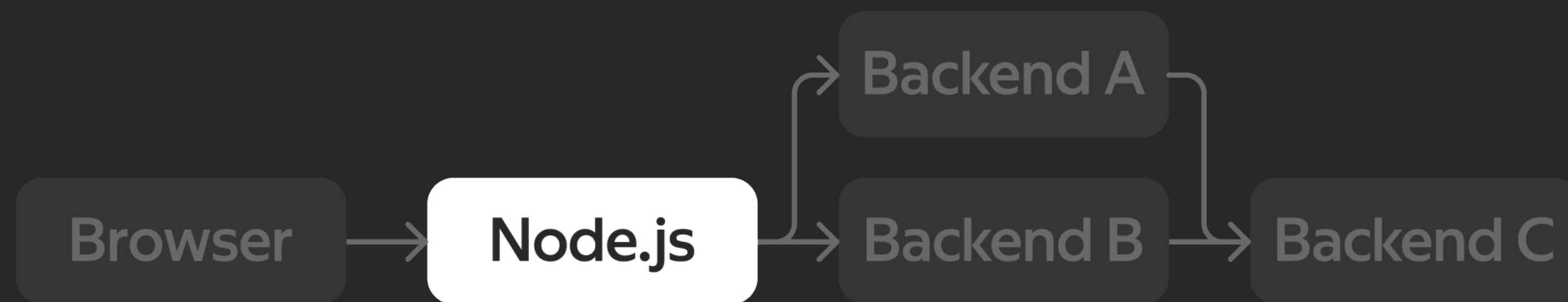
03

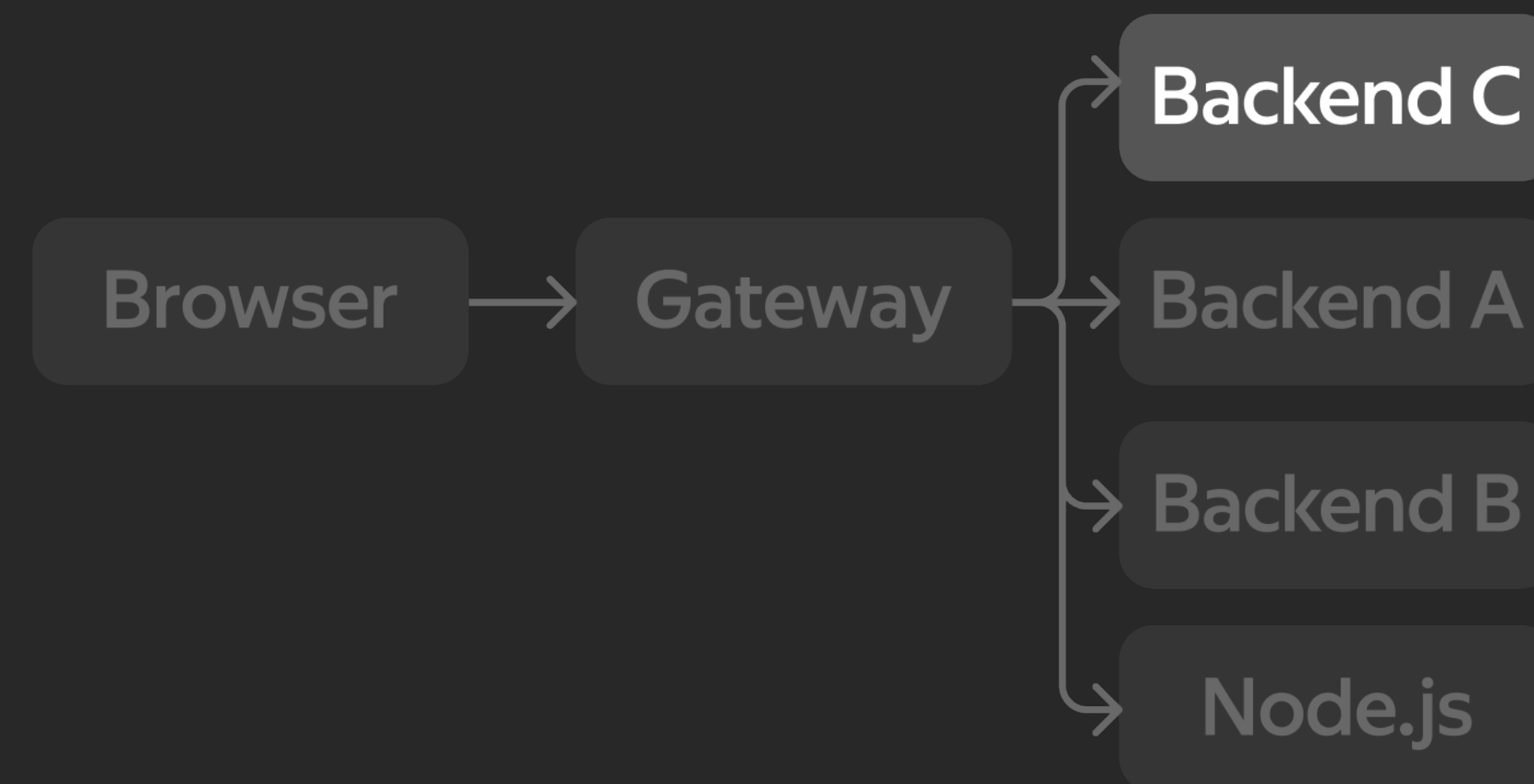
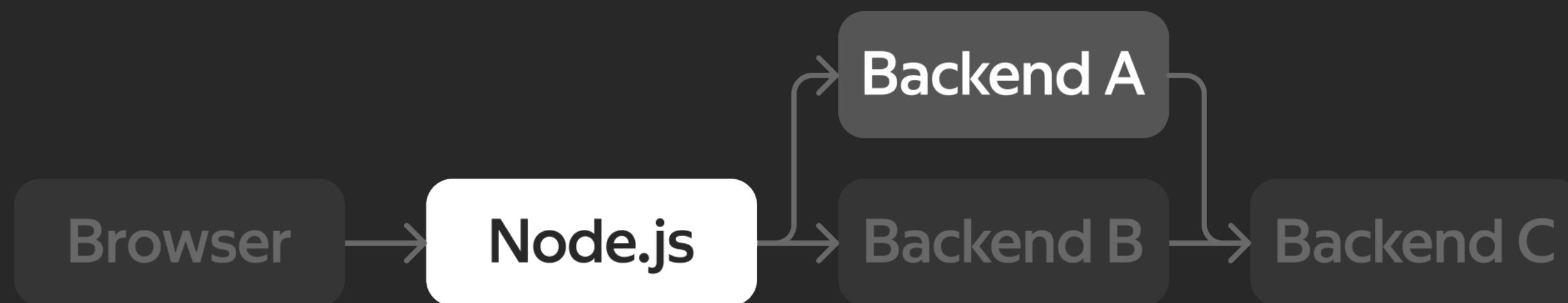
Компонент не должен содержать бизнес-логики

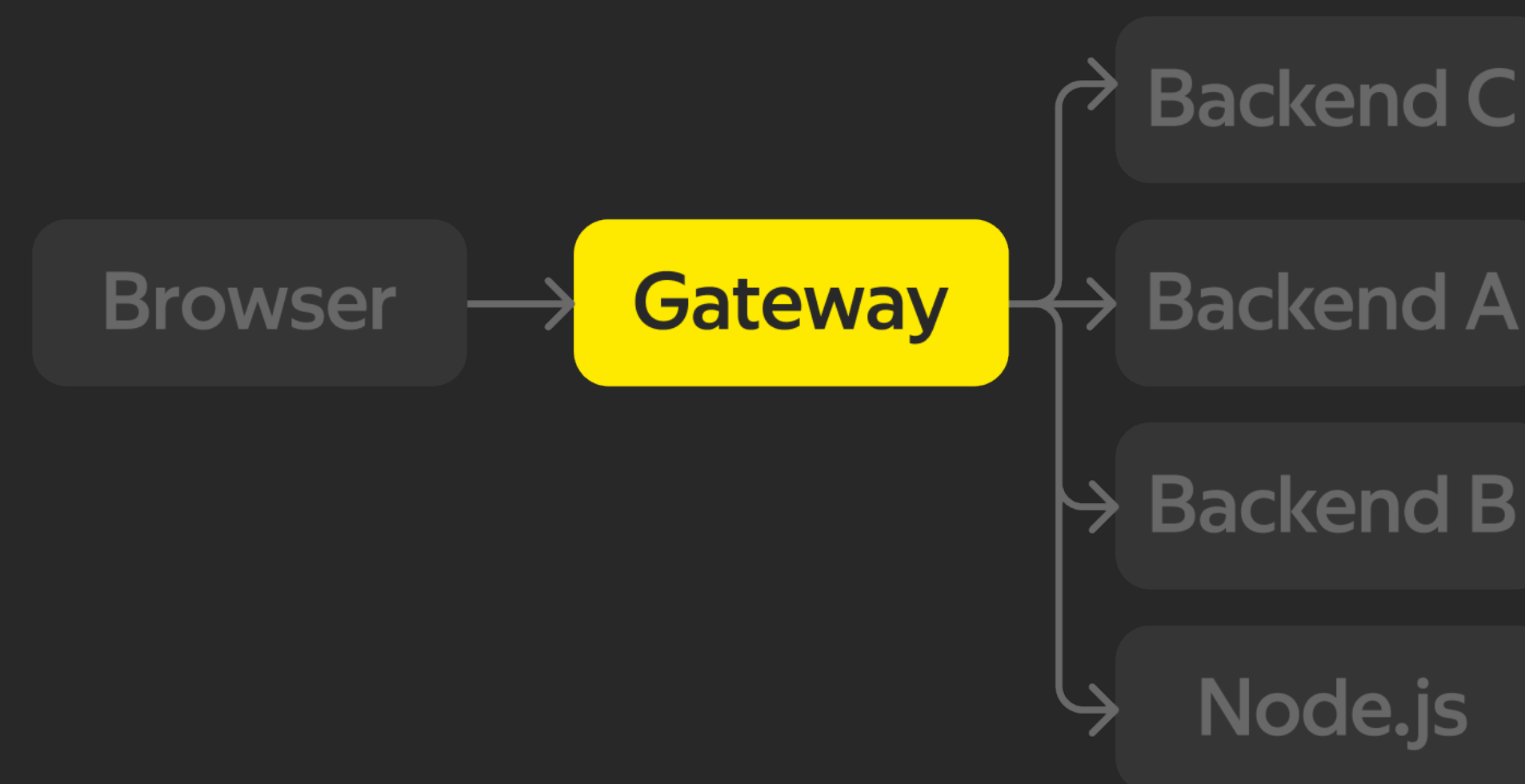
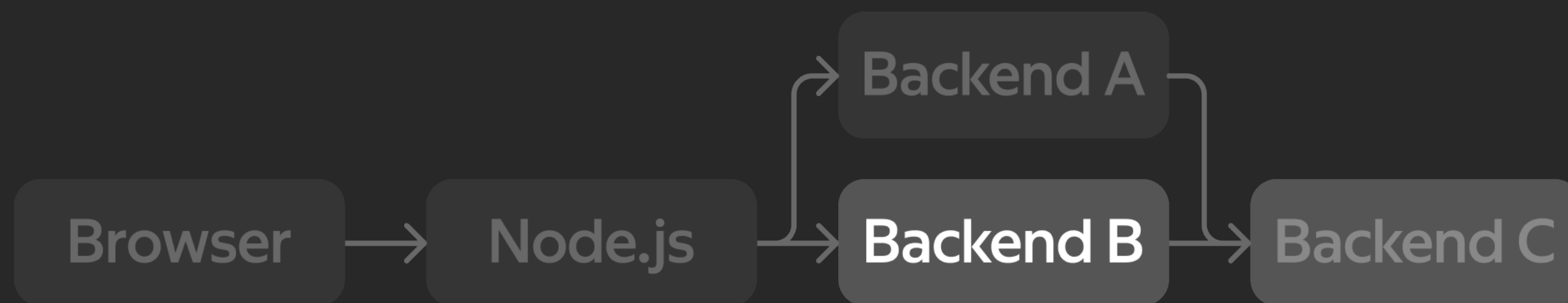


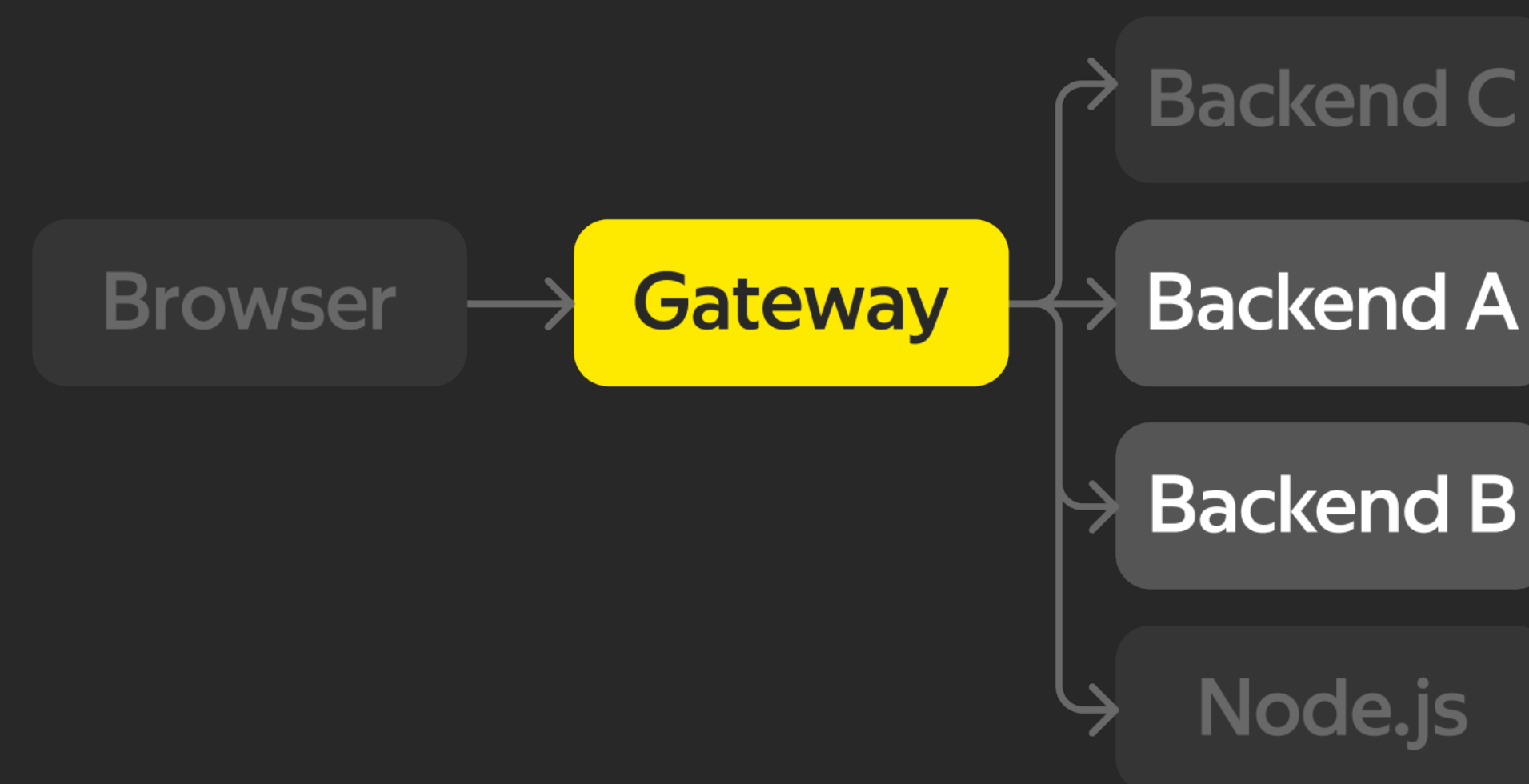
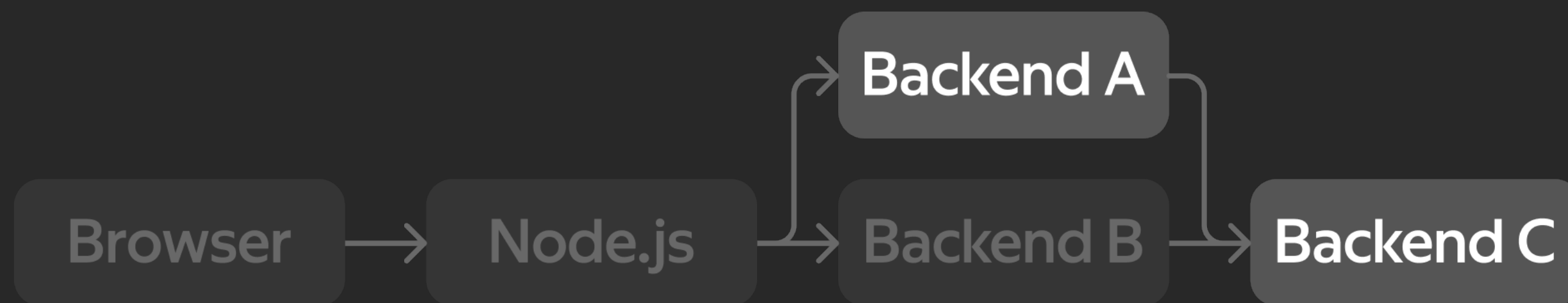




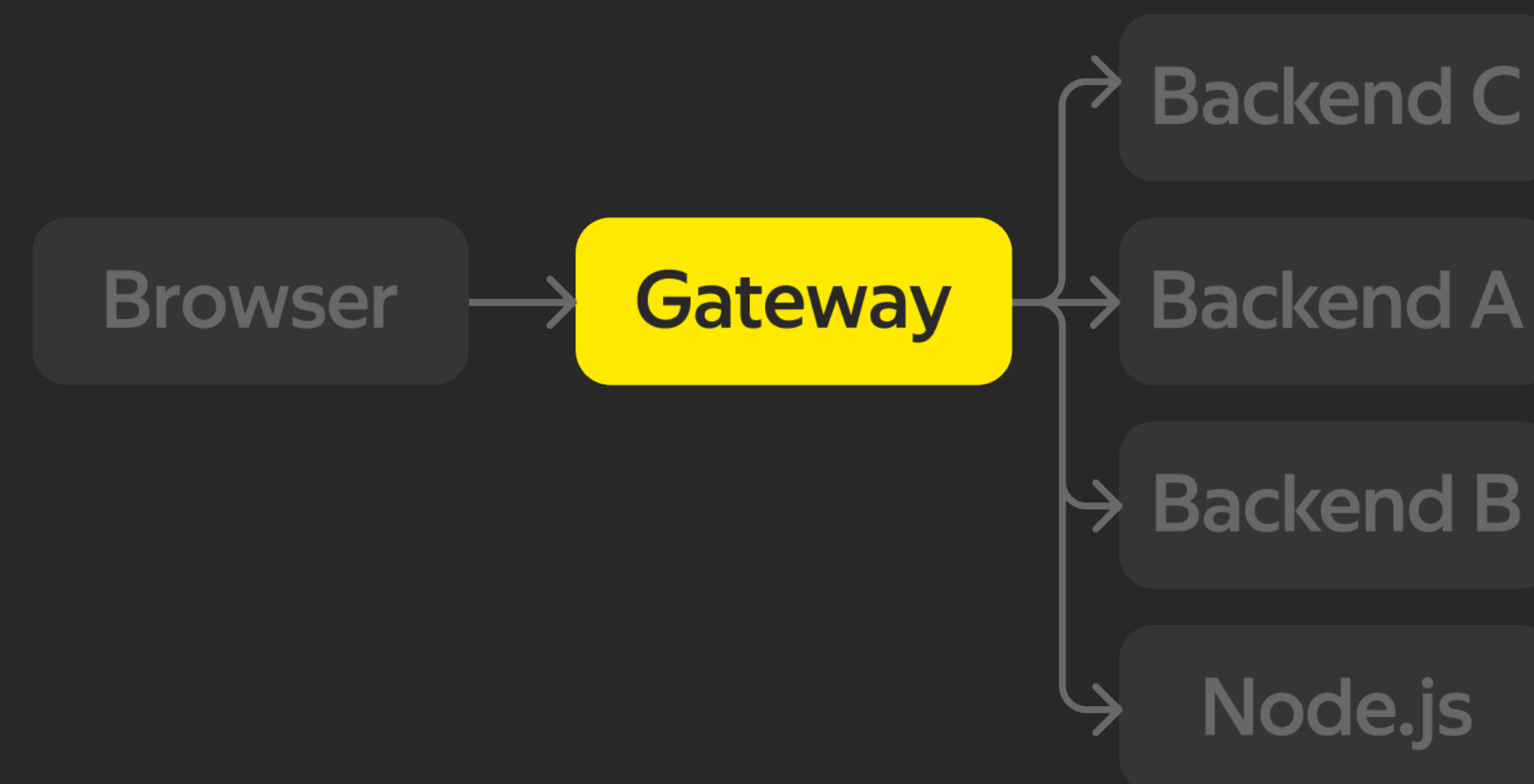
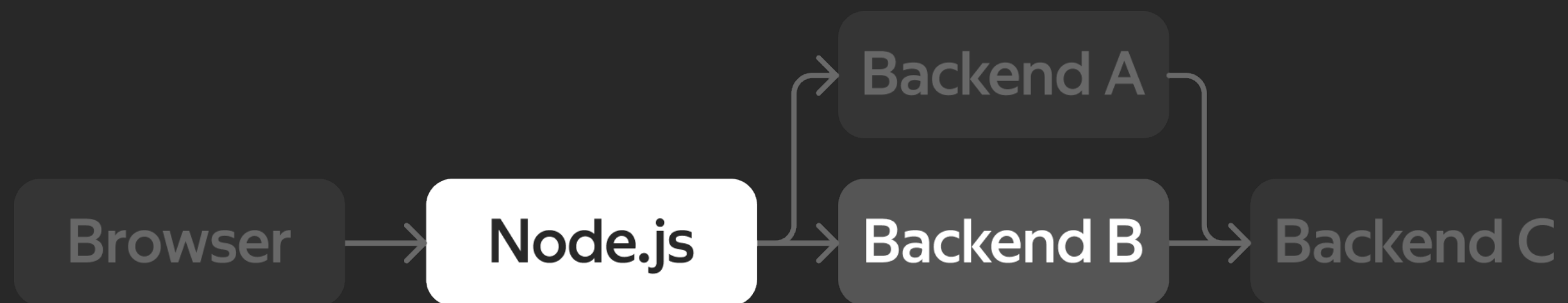


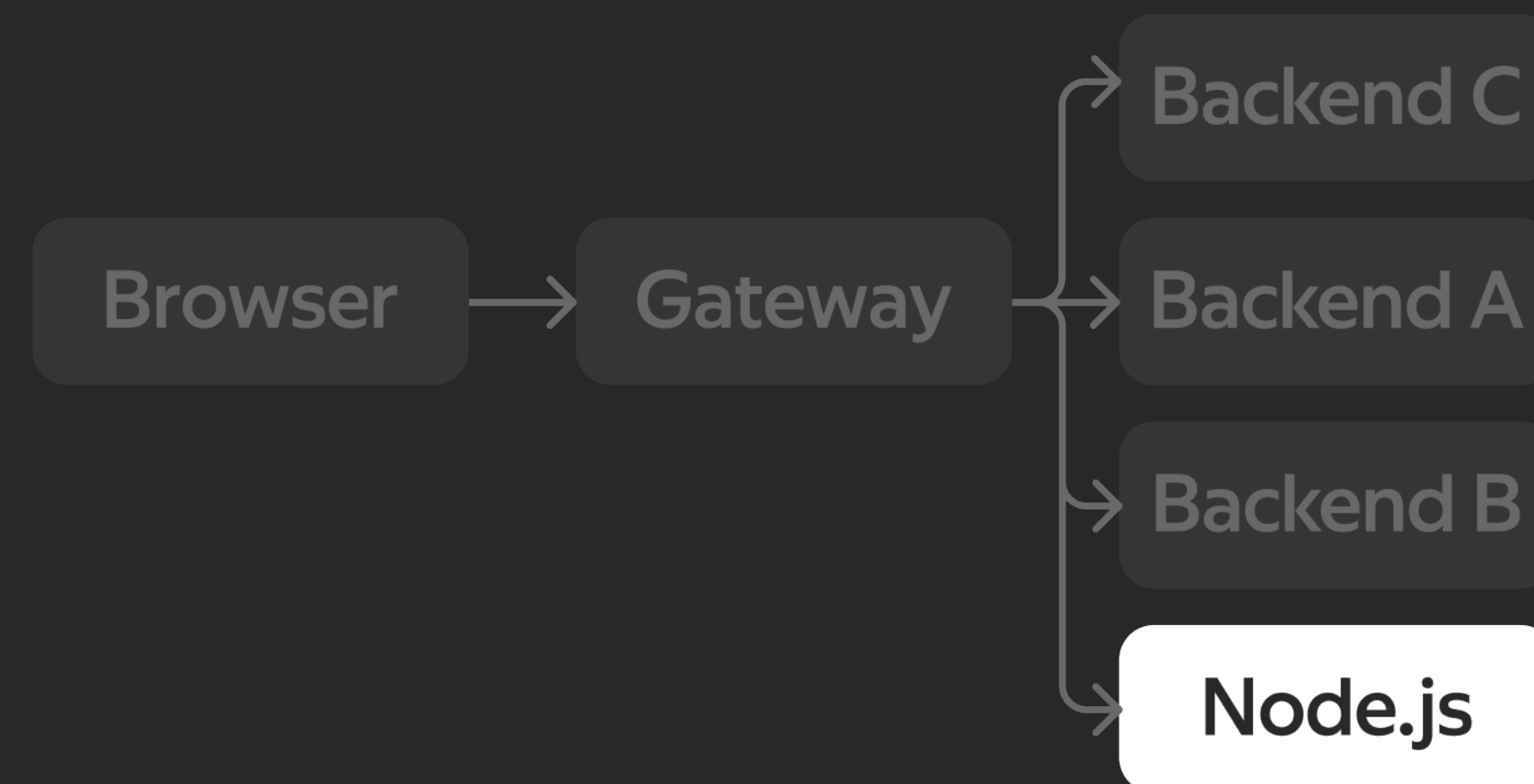
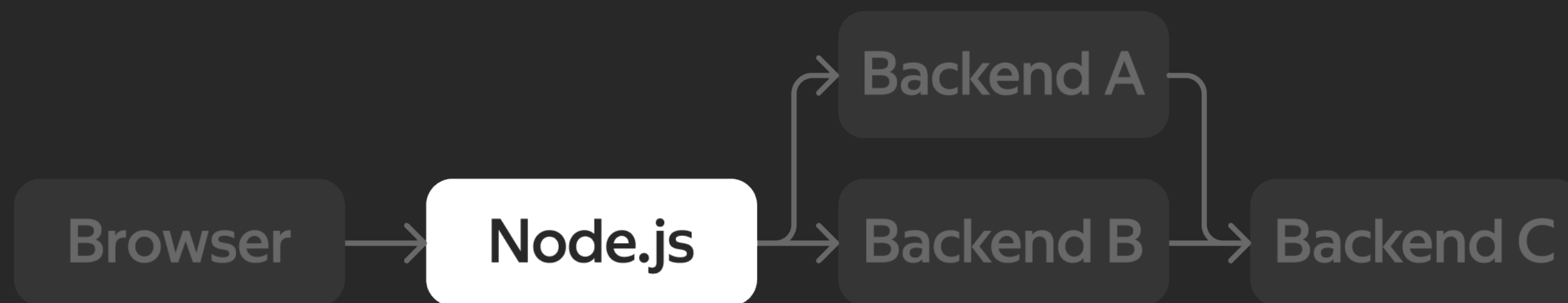


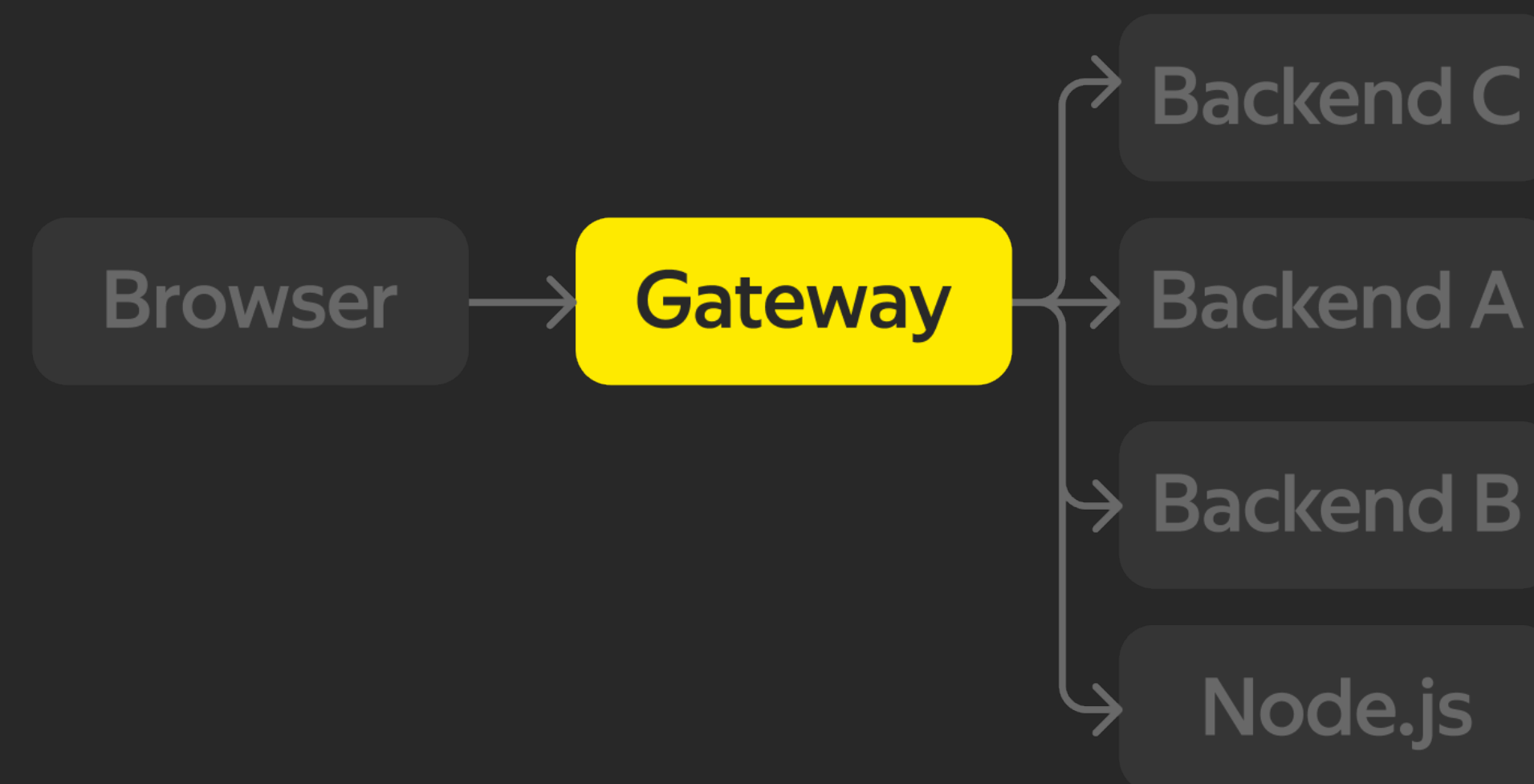
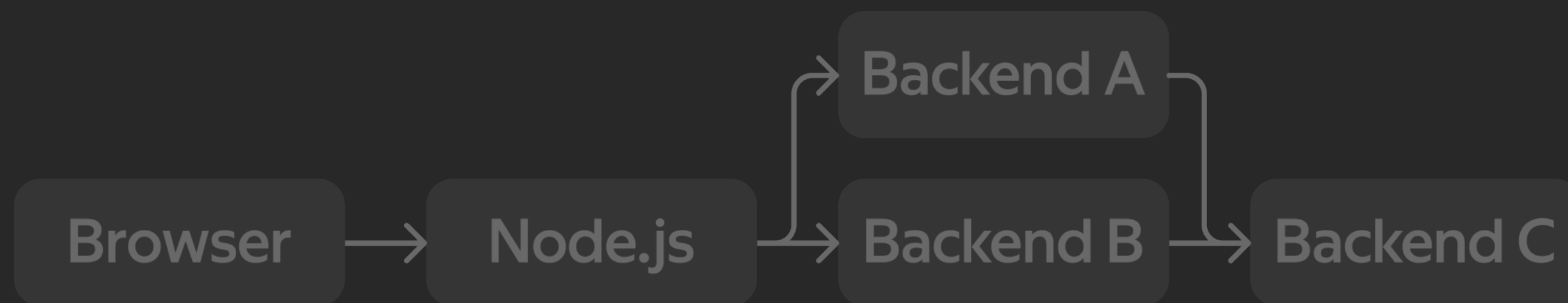














# требования

ЧТО МЫ ХОТИМ ПОЛУЧИТЬ ОТ НОВОЙ архИ

## reuse

Код работы с данными  
должен переиспользоваться  
между Web и App

## zero i/o

Node.js сервис перестаёт  
ходить по сети —  
нам не нравится налог  
на инфраструктуру

## TTFB

Должны ускориться  
в ключевых сценариях SEO

## balance

Не должны переписать  
всё в процессе

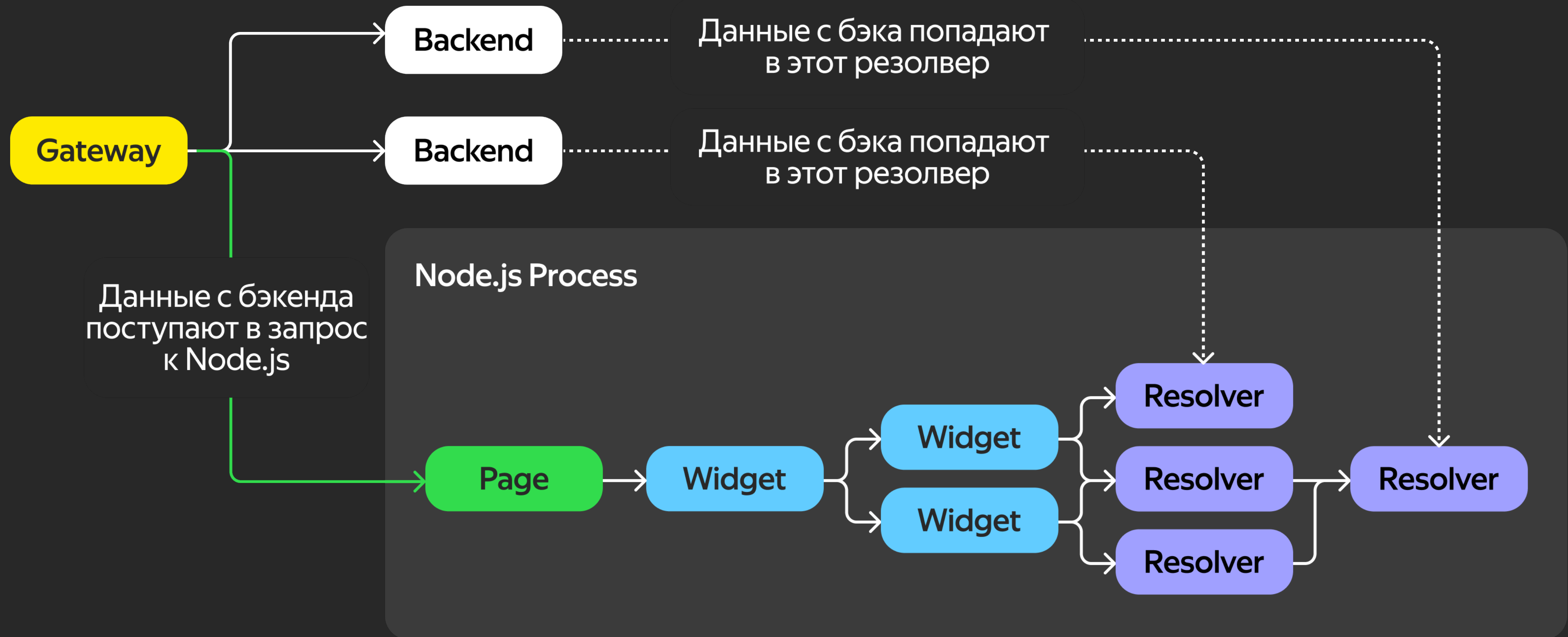
## progressive

Мы не должны потерять  
прогрессивность отдачи

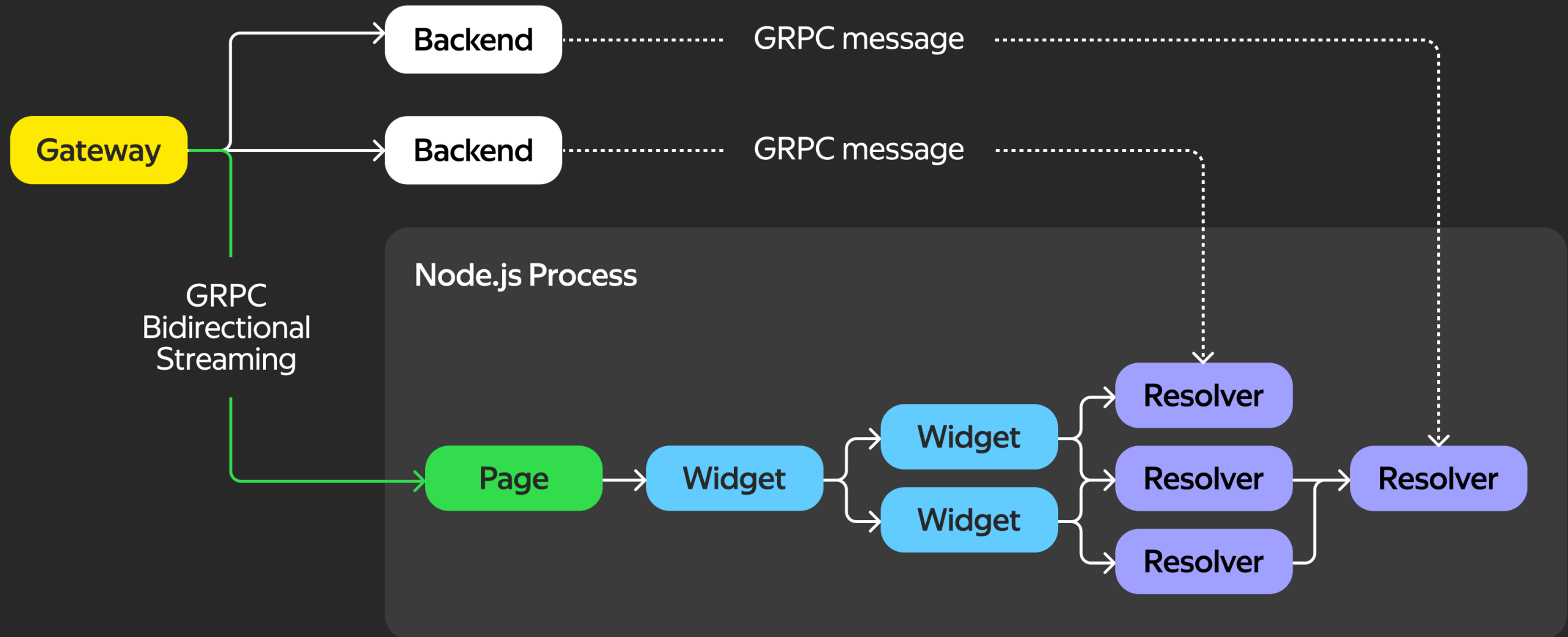
## advert

Место для вашей рекламы

# прогрессивность



# прогрессивность



# прогрессивность

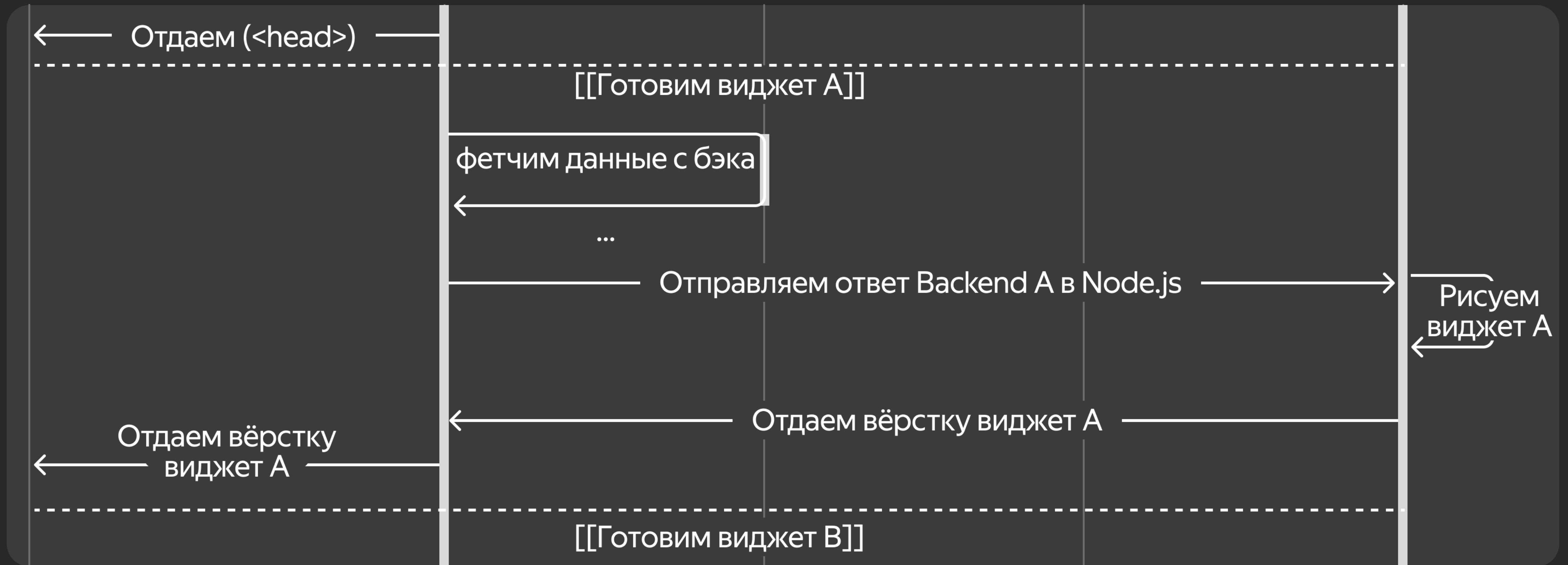


# прогрессивность

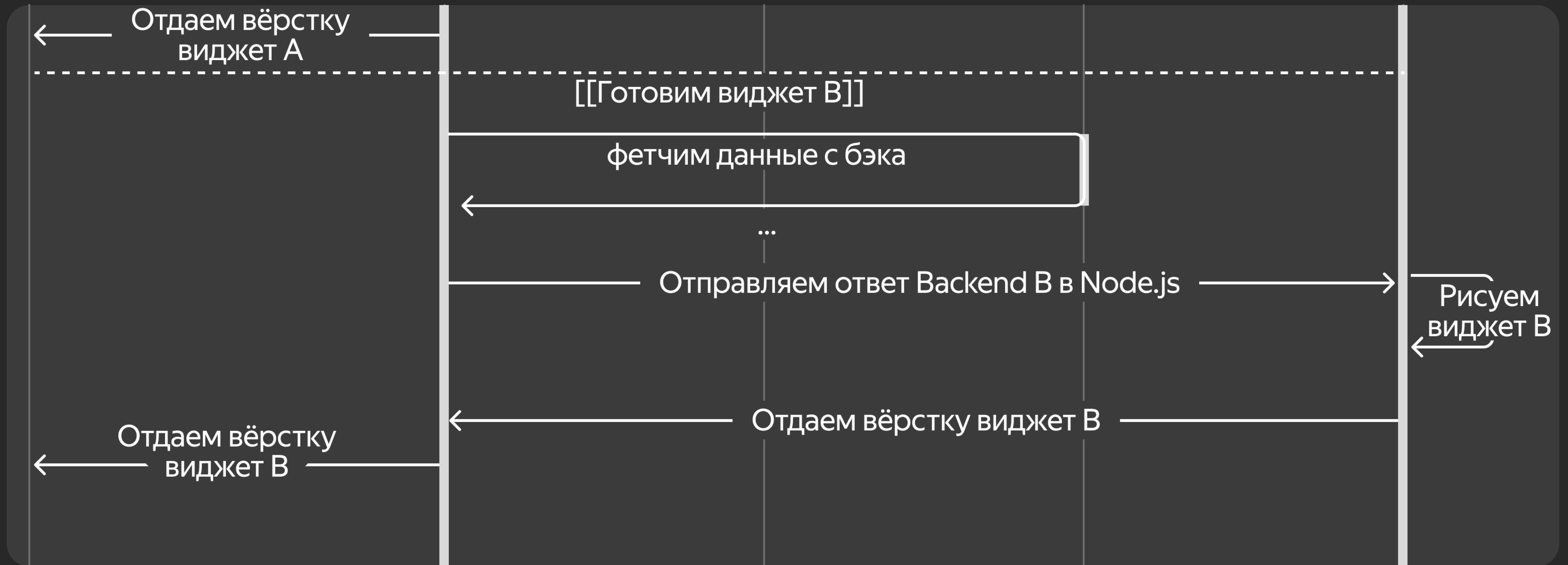




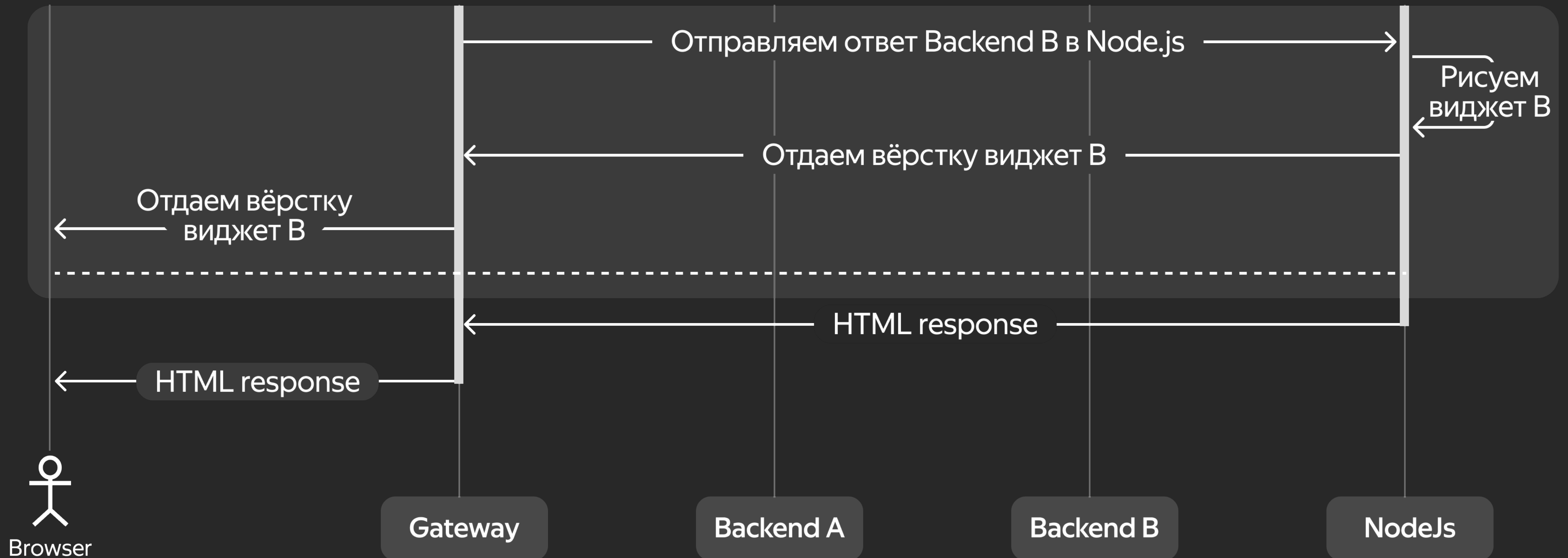
# прогрессивность



# прогрессивность



# прогрессивность



# требования

ЧТО МЫ ХОТИМ ПОЛУЧИТЬ ОТ НОВОЙ архИ

## reuse

Код работы с данными  
должен переиспользоваться  
между Web и App

## zero i/o

Node.js сервис перестает  
ходить по сети —  
нам не нравится налог  
на инфраструктуру

## TTFB

Должны ускориться  
в ключевых сценариях SEO

## balance

Не должны переписать  
всё в процессе

## progressive

Мы не должны потерять  
прогрессивность отдачи

## advert

Место для вашей рекламы

# ЧТО ПОЛУЧИЛОСЬ?

а что не очень?





# ЧТО ПОЛУЧИЛОСЬ?

01

Срезали кости  
на разработку

02

Срезали поддержку  
инфраструктуры

03

Ускорили TTFB

04

Сохранили прогрессивность  
отдачи контента

05

Автоматическое управление  
запросами

# ЧТО ПОЛУЧИЛОСЬ?

06

Полечили  
дублирование  
запросов

07

Высокий уровень  
Observability

08

Транскодинг  
протоколов  
из коробки

09

Статический анализ

10

Жёсткие контракты

# а минусы будут?

## смена парадигмы

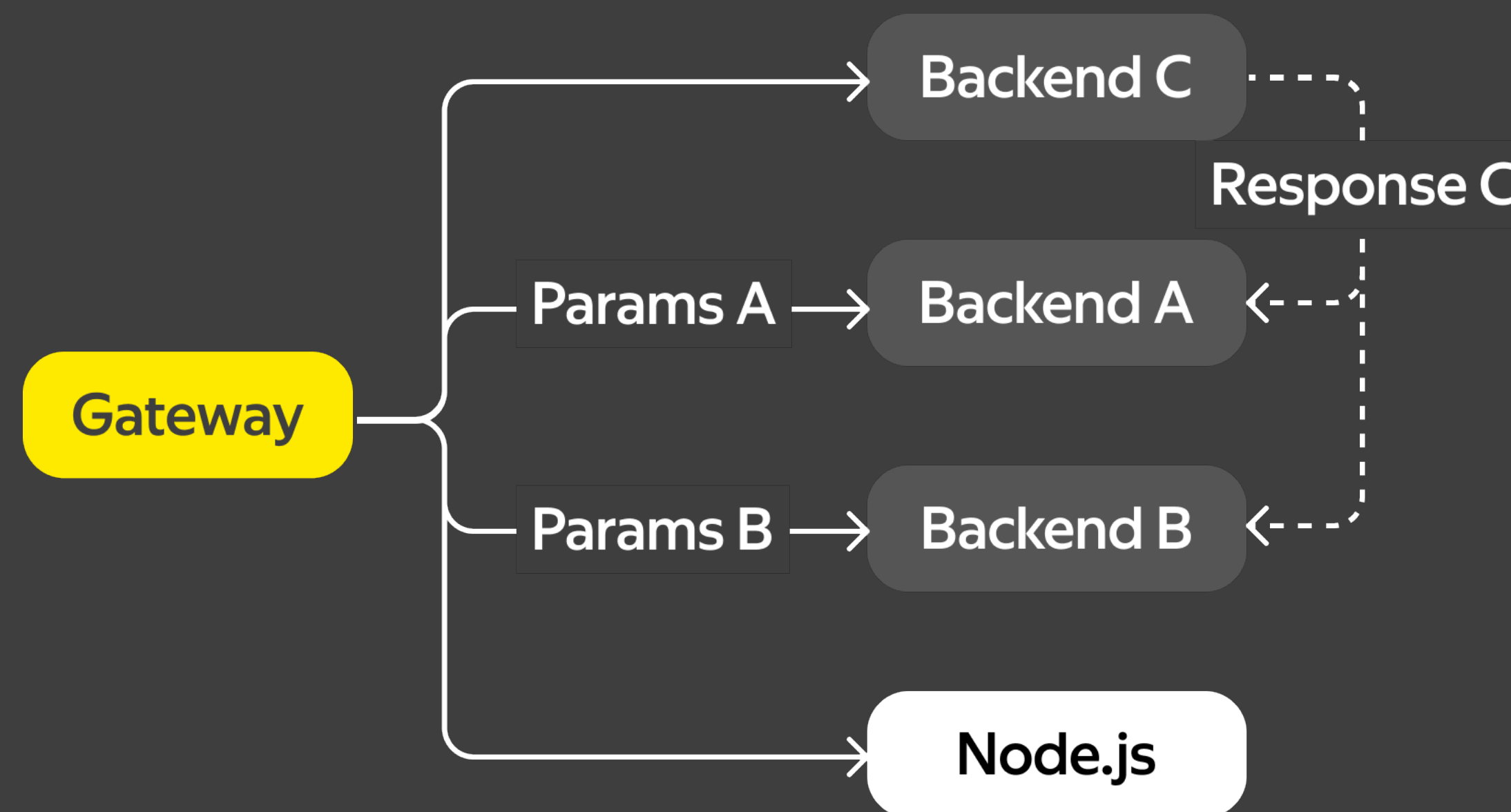
01 Когнитивная нагрузка на разработчиков

02 Как сходить из Node.JS в Gateway?

# а минусы будут?

## смена парадигмы

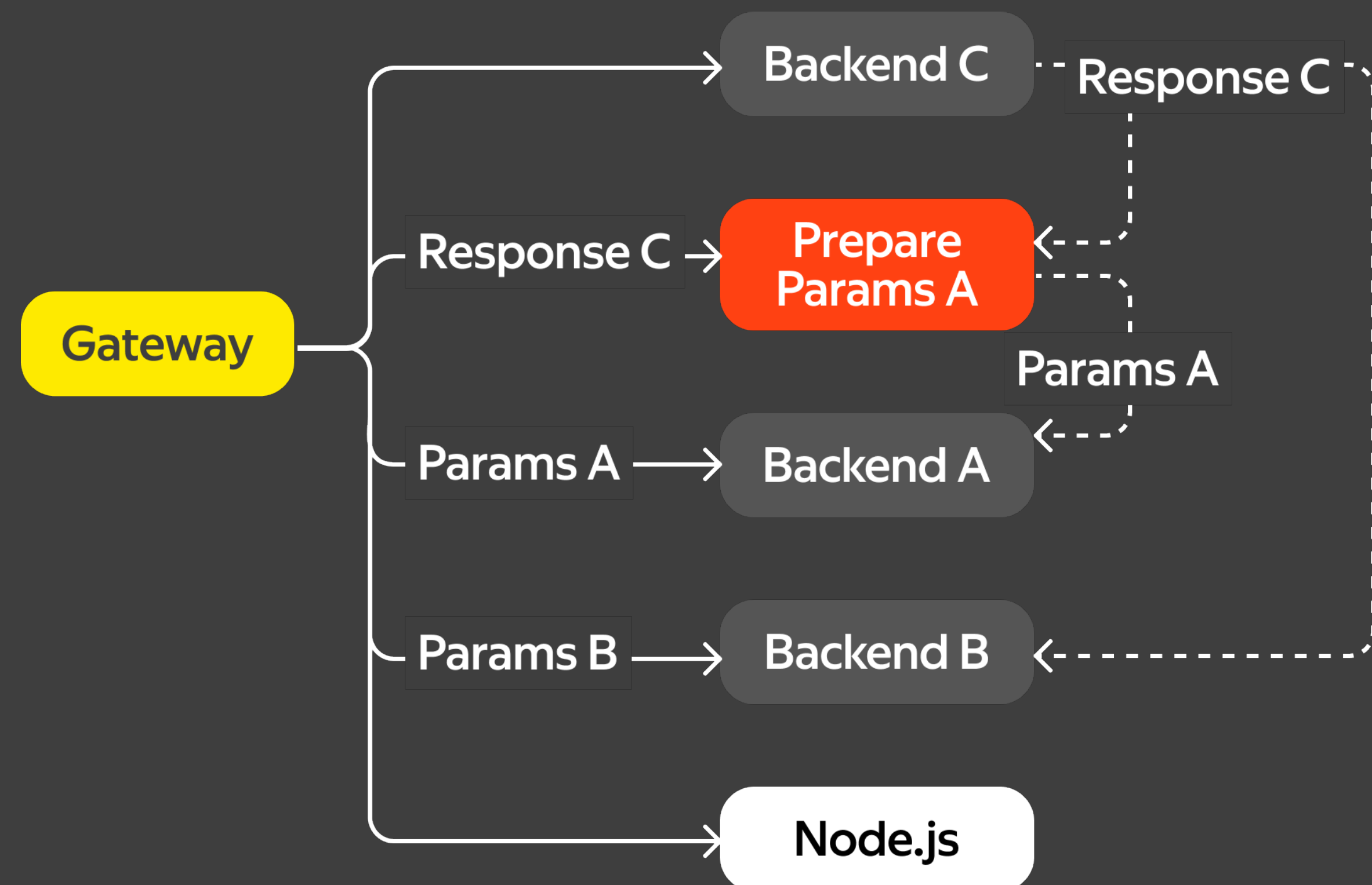
- 03 Больше ответственности на команды бэкендов
- 04 Бэкенды должны хорошо стыковаться друг с другом
- 05 Шарим контракт на троих
- 06 Подготовка запроса, чтобы сделать запрос — это запрос
- 07 Подготовка контекста должна быть общей



# а минусы будут?

## смена парадигмы

- 03 Больше ответственности на команды бэкендов
- 04 Бэкенды должны хорошо стыковаться друг с другом
- 05 Шарим контракт на троих
- 06 Подготовка запроса, чтобы сделать запрос — это запрос
- 07 Подготовка контекста должна быть общей

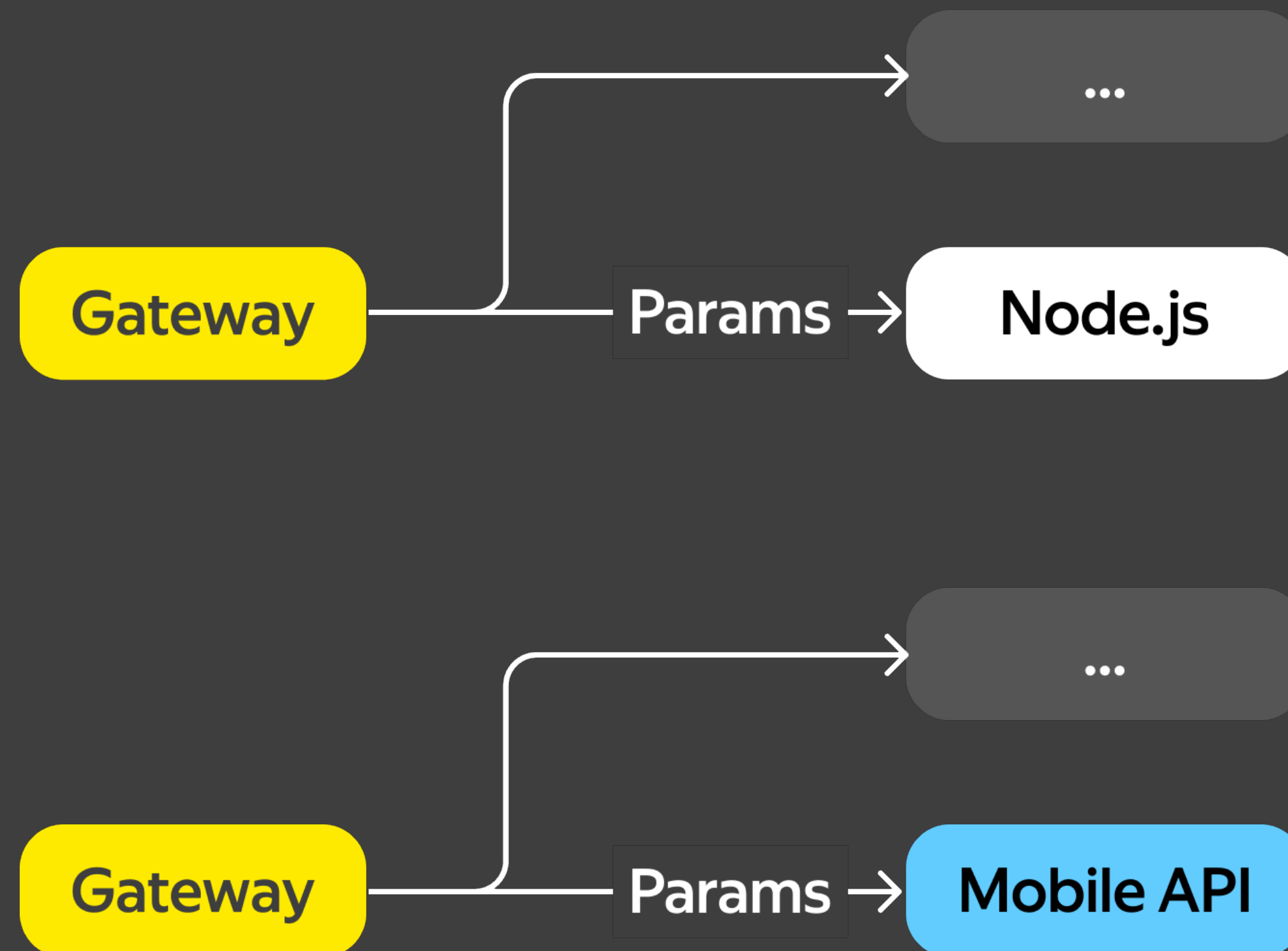


# а минусы будут?

## смена парадигмы

08

Логика работы фронтов  
должна быть похожей





# а минусы будут?

## GRPC & streaming

01

GRPC — бинарный протокол

02

В GRPC опциональные поля

03

А ещё!.. (спросите у ChatGPT)

04

Стриминг приносит сюрпризы

# а минусы будут?

## GRPC & streaming

01

Стриминг обладает меньшим количеством инвариантов

02

Стриминг растянут во времени

03

Принципиально не ретраится

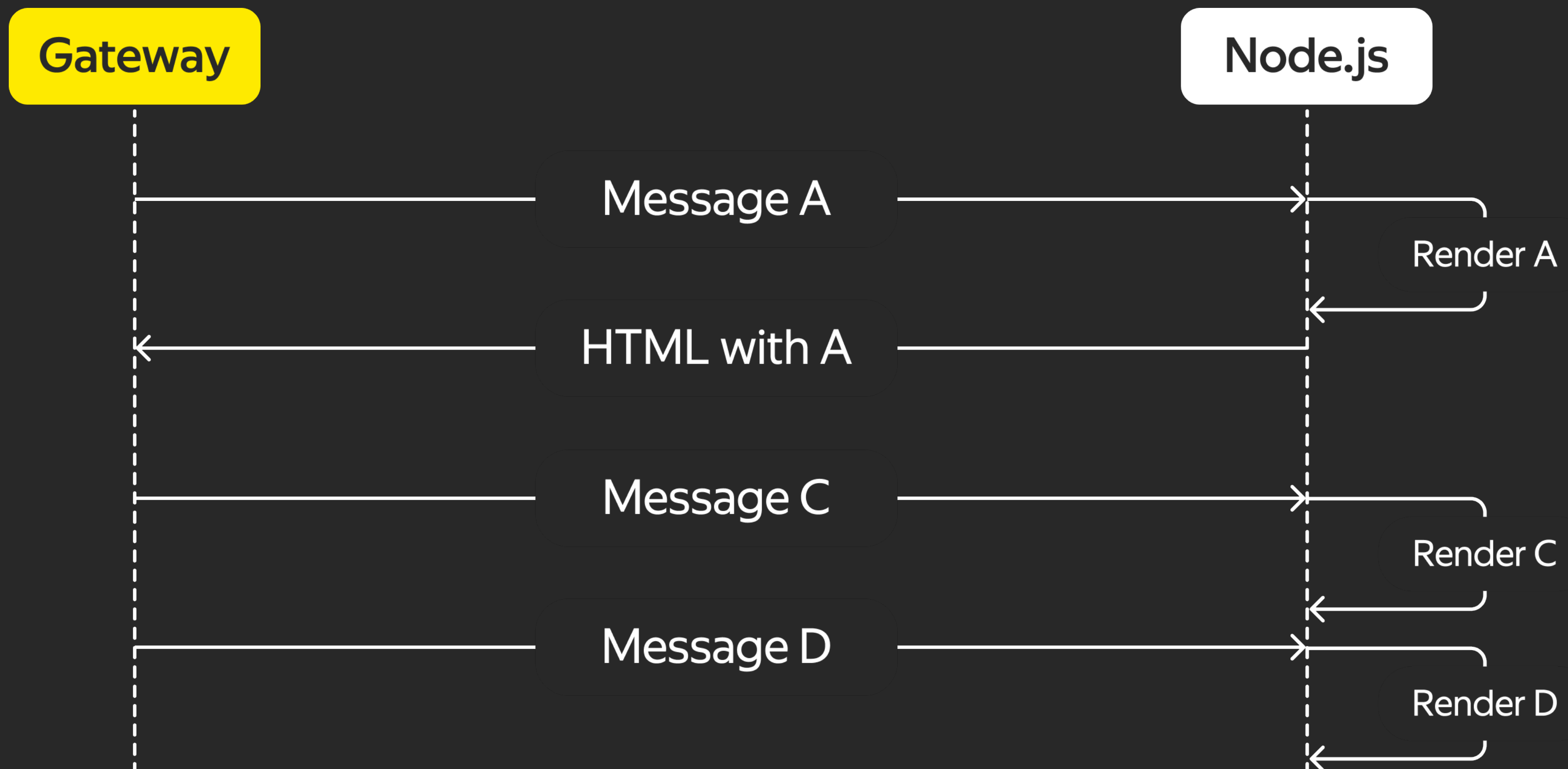
04

Запросы могут подвисать



# а минусы будут?

## GRPC & streaming



# а минусы будут?

## изменения в процессе разработки

01

Node.js зависим  
от работы Gateway

02

Зависит и в CI/CD

**решение**

поднимаем инсталляцию  
для разработки  
и тестирования

# ВЫВОДЫ



ВЫВОДЫ

01

СМОТРИТЕ  
на задачу шире



ВЫВОДЫ

02

решайте проблемы  
бизнеса, а не кода



ГОРОДСКИЕ  
СЕРВИСЫ

HolyJs

**ГОТОВ ОТВЕТИТЬ  
НА ВОПРОСЫ**

**Егор  
БЫХОВЦЕВ**

Ведущий разработчик интерфейсов

