

# Обеспечение доступности

## Технические инструменты:

Мы уже говорили про:

- Масштабирование
- Балансировка нагрузки
- Географическое распределение: размещение серверов в разных географических регионах (**CDN в том числе**)
- Кеширование
- Rate Limiting API

Еще есть инструменты:

- **Auto-Scaling**

Это процесс динамического добавления или убирания ресурсов (например, серверов, контейнеров) в зависимости от текущей нагрузки на систему.

## Как работает:

Auto-Scaling обычно базируется на определенных метриках, таких как CPU utilization, количество запросов в секунду или другие кастомные метрики. Когда эти метрики достигают определенного порога, система автоматически добавляет или убирает ресурсы.

## Варианты реализации:

1. **Horizontal Scaling:** Добавление или убирание инстансов (например, серверов).
2. **Vertical Scaling:** Увеличение или уменьшение ресурсов для конкретного инстанса (например, CPU, память).

**Пример:** Веб-сервис сталкивается с проблемой перегрузки в часы пик. Приложение начинает тормозить, что ведёт к потере клиентов.

Тогда еализуем горизонтальное автоматическое масштабирование. Как только нагрузка на один из серверов превышает 80% CPU, добавляем дополнительный инстанс (образ) сервера. После уменьшения нагрузки лишние инстансы автоматически удаляются.

- **Паттерн Service Discovery**

Service Discovery — это автоматическое обнаружение и регистрация сервисов в сети или внутри распределенной системы.

**Как работает:**

Сервисы регистрируются в централизованном реестре. Клиенты или другие сервисы могут обращаться к этому реестру, чтобы узнать, где находится нужный сервис.

**Варианты реализации:**

1. **Client-side Discovery:** Клиенты напрямую обращаются к реестру.
2. **Server-side Discovery:** Обращение к реестру происходит через специальный балансировщик нагрузки.

**Пример:** В сложной микросервисной архитектуре сложно отследить, на каком сервере или порту работает тот или иной сервис (потому что они могут автоматически масштабироваться и т.д.).

Используем Service Discovery с системой, такой как Consul. Каждый микросервис регистрирует себя при старте. Клиенты теперь могут легко находить нужные сервисы, запрашивая информацию из Consul.

- **Health Checks**

Health Checks — это проверки состояния сервисов и компонентов для определения их работоспособности.

**Как работает:**

В определенные интервалы времени или по событию выполняются проверки — например, отправка HTTP-запроса на специальный URL, и в зависимости от ответа (код состояния, время ответа и т.д.) система решает, является ли сервис здоровым.

**Варианты реализации:**

1. **Active Checks:** Система мониторинга активно отправляет запросы для проверки состояния.
2. **Passive Checks:** Система ожидает сообщений от сервисов и компонентов о их состоянии.

**Пример:** Редкие недолгие сбои в одном из микросервисов, но не понятно, в какой момент он становится недоступным.

Добавляем Health Checks, которые каждые 30 секунд проверяют доступность микросервиса. Если сервис не отвечает более чем три раза подряд, автоматически перенаправляем трафик на резервный инстанс сервера и отправляем уведомление команде поддержки.

- **Connection Pooling**

Connection Pooling — это техника повторного использования уже установленных соединений с ресурсами (например, с базой данных) вместо создания нового соединения каждый раз.

**Как работает:**

Соединения с ресурсами устанавливаются заранее и хранятся в "пуле". Когда требуется соединение, оно берется из пула, а после использования возвращается обратно.

**Пример:** Время ответа базы данных сильно варьируется, и иногда запросы занимают неприемлемо много времени.

Используем Connection Pooling для установления постоянных соединений с базой данных. Это убирает накладные расходы на установление нового соединения для каждого запроса, улучшая производительность и стабильность системы.

- **Hot/Cold Standby**

Это техника, при которой одна или несколько резервных систем готовы быстро заменить основную систему в случае ее отказа.

**Как работает:**

- **Hot Standby:** Резервные системы находятся в "горячем" режиме, синхронизируясь с основной системой и готовы к немедленному переключению.
- **Cold Standby:** Резервные системы находятся в "холодном" режиме и требуют времени на запуск и синхронизацию.

**Пример:** Система платежей критична для бизнеса, и любой её сбой может привести к значительным потерям.

Настроим Hot Standby сервер для системы платежей. Все транзакции копируются в реальном времени (из основной системы в резервную). При сбое основного сервера, система автоматически переключается на резервный, минимизируя перерыв в обслуживании.

## Организационные практики:

1. **SLA (Service Level Agreements):** Четкое определение уровней обслуживания (установление договорённостей).
2. **Мониторинг и оповещения:** Проактивный мониторинг и оповещение о событиях (когда SLA не выполняется).