Архитектура программных систем. Лекция 1 Архитектура масштабируемых веб-сервисов

Михаил Пожидаев

14 февраля 2025 г.

Сетевой ресурс

Нефункциональные требования к современному сервису

1. Обеспечение непрерывной работы с устойчивостью к сбоям и атакам.
2. Возможность проведения сервисных работ и обновлений без остановки работы сервиса.
3. Отсутствие архитектурных ограничений для обеспечения масштабирования.
4. Гетерогенность функционала, подразумевающая, что в сервисе присутствуют различные по своей природе, составу или структуре функции.

Микросервисная архитектура

Из каких блоков состоит микросервисная архитектура?

Блоки микросервисной архитектуры:

1. Балансирующий веб-сервер для принятия соединений.
2. Контейнеры с фронт-эндом для обработки запросов.
3. Брокер сообщений для организации взаимодействия контейнеров между собой.
4. База данных для хранения пользовательских данных.
5. Файловое хранилище для хранения пользовательских бинарных данных.
6. Набор микросервисов для выполнения работ.

Микросервис

Основные черты

1. Слабая связь с остальными микросервисами.
2. Изолирование фактической технологии внутри микросервиса (каждый микросервис разрабатывается своей командой).
3. Коммуникация путём сетевого взаимодействия (предпочтительно через HTTP).
4. Оформление в виде контейнера.
5. Могут находиться территориально распределённо.

Nginx

Особенности балансирующего веб-сервера

1. *Распределение нагрузки.* Балансирующий веб-сервер равномерно распределяет нагрузку между серверами фронт-энда.
2. *Масштабируемость.* Сервер автоматически адаптируется к изменениям в нагрузке, что позволяет системе справляться с ростом или уменьшением трафика.
3. *Высокая доступность.* Балансировщик обеспечивает непрерывную работу приложения, исключая из обработки сбойные серверы фронт-энда.

Front-end

Jakarta EE

1. *Управление транзакциями.* Jakarta EE предоставляет средства для управления транзакциями, которые позволяют разработчикам определять, когда несколько операций должны выполняться как единое целое.
2. *Кластеризация и масштабируемость.* Jakarta EE позволяет создавать кластеры серверов для распределения нагрузки и повышения производительности.
3. *Интеграция с базами данных.* Jakarta EE обеспечивает интеграцию с различными базами данных, такими как MySQL, Oracle, PostgreSQL и др.
4. *Контейнеры и серверы приложений.* Jakarta EE предлагает стандартизированные контейнеры и серверы, которые обеспечивают запуск и управление приложениями.

Контейнерная виртуализация

Доставка и изоляция компонентов

1. Код исполняется нативно.
2. В гостевых системах ядро общее с хост-системой.
3. Каждый контейнер имеет свой IP-адрес.
4. Лимитируется память, процессорное время и пр.

Брокер сообщений

Общение контейнеров между собой

1. *Доставка сообщений.* Брокер сообщений обеспечивает надёжную и гарантированную доставку сообщений между контейнерами, даже если некоторые из них временно недоступны или перегружены.
2. *Масштабируемость и гибкость.* Брокеры сообщений позволяют легко масштабировать систему путём добавления или удаления контейнеров в зависимости от нагрузки на систему.
3. *Очередь сообщений.* Брокеры предоставляют возможность организации очередей сообщений, что позволяет компонентам системы обрабатывать данные асинхронно.
4. *Мониторинг и отчётность.* Брокеры могут предоставлять инструменты для мониторинга и отчётности о состоянии системы и её компонентов.

Amazon S3

Хранение пользовательских файлов

1. *Экономичность.* Модель оплаты за использование (pay-as-you-go) позволяет пользователям платить только за фактически использованные ресурсы.
2. *Устойчивость к отказам.* Amazon S3 использует автоматическое восстановление после сбоев и резервное копирование данных для обеспечения устойчивости к отказам.
3. *Доступность.* Amazon S3 обеспечивает высокую доступность данных благодаря распределённой инфраструктуре и репликации данных в нескольких зонах доступности.

Спасибо за внимание!

Всё о курсе: https://marigostra.ru/materials/arch.html
E-mail: msp@luwrain.org
Канал в Телеграм: @MarigostraRu