Текст до презентації

Слайд 2

Прочитати про мету

Довгий час провідне місце в проєктуванні займала так звана “монолітна архітектура”. При даному підході вся система являє собою моноліт, який фізично розташовується на єдиній машині, запускається в одному процесі та виконує всі бізнес-операції системи.

Монолітний додаток піддається лише горизонтальному масштабуванню шляхом запуску декількох окремих серверів із кожним окремим монолітом. Але з плином часу знаходилися інші ідеї та підходи, саме таким стала сервісорієнтована архітектура (SOA), на відміну від монолітної системи, при SOA вся програма являє собою розподілену систему, яка обмінюється повідомленнями за певним протоколом. Вся система складається з набору незалежних сервісів, які фокусуються на власній задачі.

Слайд 3

Пізніше було запропоновано новий підхід до організації SOA, так звана мікросервісна архітектура (MSA). Мікросервісну архітектуру можна вважати підмножиною SOA, але все ж таки MSA відрізняється від класичного SOA. Важливим місцем для MSA є те, що кожен сервіс має мати власний обмежений контекст для цієї предметної області. Обмежень на кількість існуючих сервісів немає, але кожен сервіс має працювати лише над одною бізнес-задачею. Також всі сервіси можуть бути написані на абсолютно різних мовах програмування та використовуючи будь-які бібліотеки, також має місце децентралізоване збереження даних, тобто кожен сервіс має свою власну базу даних

Розглянемо переваги кожної з вище описаних архітектурних стилів, які представлені в таблиці

Для монолітної архітектури:

1. Простота. Даний архітектурний підхід є простішим в реалізації, управлінні та розгортанні.

2. Узгодженність. Для монолітного додатку простіше слідкувати за узгодженністю коду, опрацьовувати програмні помилки.

3. Міжмодульний рефакторинг. Єдиний кодовий репозиторій та цілісність структури полегшує роботу в ситуаціях, коли декілька модулів повинні взаємодіяти між собою або у випадку, коли необхідно перемістити деяку програмну логіку з одного модуля в інший.

Перейдемо до основних переваг SOA:

1. Часткове розгортання. Розгортання всієї системи складається з розгортання окремих сервісів-модулів, які є незалежними. У випадку монолітного додатку, при деякій зміні – нам необхідно перезбирати та знову розгортати всю систему.

2. Відмовостійкість. При виходу зі строю окремих сервісів, вся система може залишатися дієздатною за рахунок резервних модулів.

3. Відсутність стану. SOA гарантує відсутність спільних станів між модулями.

4. Гетерогенність. Незалежність та стандартизованість інтерфейсів сервісів та протоколів передачі даних дозволяють створювати сервіси використовуючи різні мови програмування та різні фреймворки.

Слайд 4

Мікросервісна архітектура вирішує проблему складності для моноліту. Переваги

Також до пееваг можна віднести:

− Стійкість системи. При виходженні з ладу системи, ми можемо локалізувати причину в рамках конкретного мікросервісу. Зі сторони програмних переваг можна виділити наступні [2]

Проте Мікросервісна архітектура не призначена для розв’язку всіх можливих задач та має властиві розподіленим системам недоліки. Найбільше труднощів виникає у питаннях взаємодії мікросервісів, їх інтеграції. Також даний термін з’явився відносно недавно, тому не існує загальноприйнятих специфікацій та рекомендацій для створення якісних додатків. Наведемо перелік основних складнощів з якими можна зіткнутися при створенні мікросервісного додатку:

− Складність розробки.

. − Збільшення використання ресурсів. Мікросервісна архітектура вимагає більше ресурсів, ніж монолітна, оскільки кожен мікросервіс необхідно забезпечити власним контейнером з розгорнутим програмним середовищем.

Збільшення навантаження на мережу. Для взаємодії мікросервіси використовують стандартні протоколи обміну мережею, коли компоненти моноліту спілкуються в рамках єдиного процесу і не вимагають додаткових мережевих викликів [4]

− Тестування , Моніторинг системи.

Слайд 5

Java – одна з найпопулярніших та найпотужніших мов програмування сучасності.

Програми на Java транслюються в байт-код, який виконується віртуальною машиною Java(JVM) – програмою, яка опрацьовує байтовий код та передає інструкції обладнанню як інтерпретатор. Перевага подібного способу виконання програм є повна незалежність байт-коду від операційної системи та апаратури, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина.

Слайд 6

Spring framework – найпотужніша Java-бібліотека, яка дозволяє створювати програмні системи будь-якої складності.

Слайд 11

Service Discovery є одним з ключових принципів мікросервісної архітектури. Задача виявлення сервісів була вирішена використанням Netflix Eureka. Eureka – це сервер для реєстрації всіх сервісів, які знаходяться в системі. Для запуску серверу я створила простий Spring Boot додаток з анотацією

При правильному налаштуванні серверу, відкрилося вікно браузера з панеллю управління