ISSN 3060-4567

SERVERLESS ARXITEKTURA VA UNI FRONT-END DASTURCHILAR UCHUN FOYDALARI

To‘xtayeva Robiya Ravshanbek qizi
Andijon davlat universiteti, talaba
tukhtayeva_robiya@adu.uz

Anotatsiya: *Mazkur maqolada serverless arxitektura konsepsiyasi va uning front-end dasturchilar uchun imkoniyatlari ko‘rib chiqiladi. Serverless arxitektura, an’anaviy serverlarni boshqarish ehtiyojini bartaraf etib, resurslarni avtomatik boshqarish orqali ilovalarni tezroq, moslashuvchan va samarali yaratish imkonini beradi. Maqolada serverless texnologiyalarining asosiy tamoyillari, ulardan foydalanishning afzalliklari va mashhur xizmatlar (AWS Lambda, Azure Functions, Google Cloud Functions) tahlil qilinadi.*

Kalit so‘zlar*: serverless arxitektura, front-end dasturchilar, AWS Lambda, Azure Functions, Google Cloud Functions, zamonaviy dasturlash*

Dasturiy ta’minot yaratish sohasida serverless arxitektura so‘nggi yillarda keng ommalashmoqda. Ushbu texnologiya dasturchilarga an’anaviy serverlarni boshqarishdan ozod bo‘lib, kod yozishga ko‘proq e‘tibor qaratishga imkon beradi. Ayniqsa, front-end dasturchilar uchun serverless arxitektura katta imkoniyatlar yaratadi, chunki u backend bilan ishlashni soddalashtiradi va mahsulotni tezroq yetkazib berish imkonini beradi.

Serverless arxitektura “serverlarsiz” ma’noni anglatsa-da, bu texnologiya serverlardan umuman voz kechishni emas, balki ularni boshqarish mas’uliyatining xizmat ko‘rsatuvchi provayder zimmasiga yuklatilishini anglatadi. Amazon Web Services (AWS), Microsoft Azure, Google Cloud kabi xizmatlar serverlarni avtomatik boshqarish, miqyoslash va texnik xizmat ko‘rsatishni o‘z zimmalariga oladi. Dasturchilar esa faqat funksionallikka e‘tibor qaratadilar.

ISSN 3060-4567

Voqea asoslangan hisoblash (Event-driven computing): Serverless xizmatlari asosan voqea (event) paydo bo‘lishiga asoslanadi. Masalan, foydalanuvchi ma‘lumot yuborganda yoki fayl yuklanganda funksiyalar avtomatik ishga tushadi. Foydalanuvchi faqatgina xizmatlardan foydalanish davomida to‘lov qiladi, ya’ni doimiy server ishlashi uchun resurslar to‘lanmaydi. Yozilgan kod talabga qarab avtomatik miqyoslanadi va bir vaqtning o‘zida bir nechta mijozlarga xizmat ko‘rsata oladi. Amazon’ning serverless hisoblash xizmati bo‘lib, voqeaga asoslangan funksiyalarni bajarishga mo‘ljallangan. Google tomonidan taqdim etiladigan xizmat bo‘lib, dasturlarning backend qismini boshqarishni yengillashtiradi. Microsoft tomonidan ishlab chiqilgan ushbu xizmat kodni voqealar asosida ishga tushirishga imkon beradi.

Backendni boshqarishdagi murakkabliklarni kamaytirish: Front-end dasturchilar uchun backendni boshqarish va sozlash ko‘p vaqt va resurs talab qiladi. Serverless arxitektura yordamida bu jarayonlar provayderlar tomonidan avtomatlashtiriladi, bu esa dasturchilarga faqat interfeys va biznes logikasini rivojlantirishga imkon beradi. Mahsulotni tezkor yetkazib berish: Serverless texnologiyalar orqali prototiplash va funksionallikni ishlab chiqish tezlashadi. Chunki dasturchilar serverlarni sozlash va boshqarish bilan chalg‘imaydilar. Moslashuvchanlik va miqyoslash: Serverless arxitektura talabga qarab resurslarni avtomatik oshiradi yoki kamaytiradi. Bu, ayniqsa, katta trafik oqimiga ega bo‘lgan ilovalar uchun muhimdir. Arzonlashgan xarajatlar: Front-end dasturchilar uchun serverlarni boshqarish xarajatlarini qisqartirish katta afzallikdir. Serverless xizmatlar faqat ishlatilgan vaqt uchun to‘lov qilishga imkon beradi. Funktsional modullar bilan ishlash: Serverless arxitektura modul asosida ishlashni qo‘llab-quvvatlaydi. Har bir modul yoki funksiya mustaqil ravishda yoziladi va ishlaydi. Bu, kodning qayta ishlatilishini osonlashtiradi.

Masalan, foydalanuvchi tomonidan yuklangan rasmlarni o‘lchamlarini avtomatik o‘zgartirish yoki ma‘lumotlarni tahlil qilish. Chat ilovalarida real vaqtda xabarlarni yetkazib berish uchun. Serverless yordamida tezkor va xavfsiz API-lar yaratish mumkin. Fayllarni yuklash, saqlash va boshqarish uchun serverless xizmatlar

ISSN 3060-4567

juda qulaydir. IoT qurilmalardan ma’lumotlarni yig'ish va tahlil qilish uchun serverless xizmatlar juda mos keladi. Mobil ilovalar uchun serverless texnologiya ishlatish natijasida resurslar samarali boshqariladi va dastur barqaror ishlaydi.

Serverless arxitektura front-end texnologiyalari bilan yaxshi integratsiya qilinadi. Masalan, React, Angular yoki Vue ilovalari serverless backend bilan ishlash uchun GraphQL yoki REST API orqali bog‘lanadi. Bundan tashqari, Firebase kabi xizmatlar front-end dasturchilarga to‘liq backendni boshqarishga ehtiyoj qoldirmaydi. Serverless texnologiyalarning SDK yoki kutubxonalari (masalan, AWS SDK, Firebase SDK) yordamida front-enddan serverless funksiyalarni chaqirish juda osonlashadi.

Serverless xizmatlar odatda ma‘lum vaqt oralig‘ida funksiyani bajarishga ruxsat beradi (masalan, AWS Lambda uchun 15 daqiqa). Monitoring murakkabligi: Serverless arxitekturada xatolarni kuzatish va tahlil qilish qiyinroq bo‘lishi mumkin. Provayderga bog‘liqlik: Ma‘lumotlar va xizmatlar provayder platformasiga bog‘liq bo‘lib qolishi mumkin.

Global miqyosdagi joylashuv: Serverless provayderlar xizmatlarni dunyo bo'ylab taqsimlaydi, bu esa foydalanuvchilarga qisqa vaqt ichida xizmat ko'rsatishni ta'minlaydi. Xavfsizlikni boshqarish: Serverless platformalar xavfsizlikning ko'p qatlamlarini ta'minlaydi, shu jumladan autentifikatsiya va ma'lumotlarni shifrlash. Monitoring va tahlil: CloudWatch, Firebase Analytics kabi xizmatlar serverless arxitektura orqali real vaqt rejimida monitoring imkoniyatlarini beradi. Zaxiralash va tiklash: Serverless arxitekturada avtomatik zaxiralash tizimlari mavjud bo'lib, ular ma'lumotlarni yo'qotish xavfini kamaytiradi. Front-end dasturchi sifatida, siz faqat backendni ishlatib, undan ma'lumot olish uchun API chaqirishingiz mumkin. Masalan, AWS Lambda yordamida oddiy serverless API yarataylik: Lambda function, foydalanuvchi so'rovini oladi va ma'lumotlarni qaytaradi.

ISSN 3060-4567

```
import React, { useState, useEffect } from 'react';

function App() {
  const [message, setMessage] = useState('');

  useEffect(() => {
    // Serverless API endpoint
    fetch('https://your-api-endpoint.amazonaws.com/dev/hello')
      .then((response) => response.json())
      .then((data) => setMessage(data.body));
  }, []);

  return (
    <div className="App">
      <h1>{message}</h1>
    </div>
  );
}
```

Serverless arxitektura front-end dasturchilari uchun katta qulaylik yaratadi. Backendni boshqarishga hojat yo'q, chunki serverlarni bulut platformalari avtomatik ravishda boshqaradi. Bu front-end dasturchilariga tezda API-lar yaratish, ma'lumotlar olish va samarali prototiplar ishlab chiqish imkonini beradi. Serverless platformalar avtomatik miqyoslashni ta'minlab, tizim samaradorligini oshiradi va xarajatlarni kamaytiradi. Shunday qilib, front-end dasturchilari backendni boshqarishga vaqt sarflamasdan, asosiy ishlariga e'tibor qaratishlari mumkin.

FOYDALANILGAN ADABIYOTLAR:

1. React Native Documentation – React Native rasmiy hujjatlari. Onlayn manba
2. Expo Documentation – Expo rasmiy hujjatlari, React Native uchun rivojlantirish muhitini taqdim etuvchi platforma. Onlayn manba
3. Brown, A. (2021). React Native Cookbook: Bringing the Web to Native Platforms. Packt Publishing.
4. Chinnathambi, A. (2020). React Native for Mobile Development: Harness the Power of React Native to Create Stunning iOS and Android Applications. Apress.
5. Kumar, N. (2019). Mastering React Native. Packt Publishing.