| **Nama Tools** | **Kekurangan** | **Kelebihan** |
| --- | --- | --- |
| 1. Apache Bench (ab) | Sangat Dasar: Tidak mendukung pengujian kompleks atau skenario pengujian yang beragam.  Tidak Mendukung Distribusi Beban: Hanya dapat menjalankan pengujian dari satu mesin. | Ringan dan Cepat: Sangat sederhana dan dapat digunakan langsung dari CLI.  Mudah Digunakan: Tidak membutuhkan konfigurasi kompleks, hanya perlu beberapa argumen.  Ideal untuk HTTP Benchmark: Cocok untuk pengujian sederhana terhadap server web. |
| 2. k6 | Fitur Terbatas untuk Non-HTTP Protocols: Fokus pada pengujian API dan protokol HTTP, tidak sekomprehensif JMeter dalam mendukung protokol seperti FTP, JDBC, dan lain-lain.  Kurang Grafis: Tidak menyediakan antarmuka pengguna seperti JMeter untuk pengguna non-teknis. | Berbasis JavaScript: Mendukung pembuatan skenario pengujian menggunakan JavaScript, yang sangat familiar bagi banyak developer.  Ringan dan Cepat: Dirancang untuk pengujian modern dengan konsumsi sumber daya yang minimal.  Integrasi CI/CD yang Baik: Cocok untuk pengujian berkelanjutan (continuous testing) dalam pipeline DevOps.  Open Source dan Cloud-ready: Mendukung pengujian berbasis cloud dengan k6 Cloud. |
| 3. Locust | Kurang Mendalam: Tidak memiliki fitur yang kaya seperti JMeter untuk pengujian protokol non-HTTP.  Antarmuka Terbatas: Lokust lebih berfokus pada penggunaan berbasis skrip daripada antarmuka visual. | Berbasis Python: Skenario pengujian dapat ditulis dalam Python, memudahkan pengguna yang sudah familiar dengan bahasa tersebut.  Ringan dan Mudah Digunakan: Locust dirancang untuk pengujian yang ringan dan efisien, terutama untuk skenario pengujian HTTP sederhana.  Distribusi Beban Mudah: Dapat dengan mudah menjalankan pengujian terdistribusi untuk beban yang lebih besar. |
| 4. Gatling | Kurva Belajar: Membutuhkan pengetahuan dasar tentang Scala, yang mungkin menyulitkan bagi pengguna non-programmer.  Dokumentasi Kurang Lengkap: Beberapa fitur memiliki dokumentasi terbatas dibandingkan JMeter. | User-friendly DSL (Domain Specific Language): Gatling menggunakan DSL berbasis Scala yang memudahkan pembuatan skenario pengujian secara programatis.  Kinerja Tinggi: Dibangun di atas arsitektur asinkron berbasis Netty, memungkinkan pengujian beban besar dengan konsumsi sumber daya minimal.  Laporan Mendalam: Menyediakan laporan yang interaktif dan informatif dalam bentuk grafis.  Integrasi CI/CD: Mendukung integrasi dengan pipeline CI/CD seperti Jenkins. |