**LAPORAN EKSPERIMEN MENGENAI PROBLEM KEGAGALAN MENGGUNGGAH FILE BERUKURAN BESAR**

*Laporan ini disusun untuk memenuhi tugas 3 mata kuliah Pengembangan Web (Teori)*



Disusun oleh kelompok B4:

Asri Husnul Rosadi 221524035

Faris Abulkhoir 221524040

Mahardika Pratama 221524044

Muhamad Fahri Yuwan 221524047

Najib Alimudin Fajri 221524053

Sarah 221524059

Septyana Agustina 221524060

Dosen Pengampu:

Joe Lian Min, M.Eng.

**JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA**

**PROGRAM STUDI D4 TEKNIK INFORMATIKA**

**POLITEKNIK NEGERI BANDUNG**

**2024**

# **DAFTAR ISI**

[DAFTAR ISI i](#_Toc175471873)

[A. IDENTIFIKASI PROBLEM 1](#_Toc175471874)

[B. DESKRIPSI PROBLEM 1](#_Toc175471875)

[C. METODOLOGI EKSPERIMEN 1](#_Toc175471876)

[D. PELAKSANAAN EKSPERIMEN 2](#_Toc175471877)

[E. ANALISIS HASIL EKSPERIMEN 3](#_Toc175471878)

# **IDENTIFIKASI PROBLEM**

Dalam sebuah aplikasi web, proses upload file adalah salah satu fitur yang umum digunakan. Namun, ketika berhadapan dengan file berukuran besar, seringkali terjadi masalah seperti **timeout** atau **gagal upload** akibat batasan ukuran file atau durasi eksekusi pada server. Masalah ini perlu diatasi agar aplikasi dapat menangani pengunggahan file besar secara andal dan efisien.

# **DESKRIPSI PROBLEM**

Saat mengunggah file besar ke server Node.js, beberapa masalah yang sering terjadi meliputi:

* **Timeout**: Jika file terlalu besar dan server tidak dapat menyelesaikan proses upload dalam waktu yang ditentukan, proses akan berhenti dan file tidak terunggah.
* **Keterbatasan Ukuran File**: Beberapa middleware atau konfigurasi server membatasi ukuran maksimal file yang bisa diunggah, yang menyebabkan file besar tidak dapat diproses.
* **Ketidakstabilan Koneksi**: Saat koneksi tidak stabil, proses upload dapat terganggu, menyebabkan seluruh file harus diunggah ulang dari awal, yang memakan waktu dan bandwidth.

Dua solusi yang mungkin diterapkan untuk mengatasi masalah ini adalah:

1. Meningkatkan batas maksimal ukuran file dan timeout pada server.
2. Menggunakan chunked uploads, di mana file besar dipecah menjadi beberapa potongan kecil yang diunggah secara terpisah dan digabungkan kembali di server.

# **METODOLOGI EKSPERIMEN**

Untuk mengatasi masalah ini, dilakukan eksperimen dengan dua pendekatan:

1. **Meningkatkan batas maksimal dan timeout** pada server Node.js untuk memungkinkan pengunggahan file besar secara langsung.
2. **Mengimplementasikan chunked uploads** di mana file besar dipecah menjadi beberapa bagian kecil (chunks) yang diunggah secara terpisah dan digabungkan kembali di server.

Setiap solusi akan diuji dengan menggunakan alat seperti **Postman** untuk mengunggah file besar dan melihat apakah proses upload berhasil atau tidak. Parameter seperti waktu upload, penggunaan sumber daya, dan ketahanan terhadap kegagalan akan dievaluasi.

# **PELAKSANAAN EKSPERIMEN**

**Solusi 1: Meningkatkan Batas Maksimal dan Timeout pada Server**

1. **Setup Server**:
   * Dibuat server Node.js dengan menggunakan **Express** dan **Multer** sebagai middleware untuk menangani file uploads.
   * Batas maksimal ukuran file dan waktu timeout ditingkatkan untuk memungkinkan upload file besar.
   * **Server dijalankan** di http://localhost:3000.
2. **Uji Coba dengan Postman**:
   * Dilakukan pengujian dengan mengunggah file berukuran 90MB menggunakan Postman.
   * **Observasi hasil** untuk memastikan apakah file berhasil diunggah tanpa mengalami timeout atau error.

**Solusi 2: Menggunakan Chunked Uploads**

1. **Setup Server dengan Dukungan Chunked Uploads**:
   * Server Node.js diatur untuk menangani chunked uploads dengan menggunakan **Multer** untuk menyimpan chunk secara sementara dan **Node.js File System (fs)** untuk menggabungkan chunk di server.
   * **Server dijalankan** di http://localhost:3000.
2. **Uji Coba dengan Postman**:
   * Menggunakan Postman untuk mengunggah file dalam beberapa chunk. Setiap chunk diunggah sebagai request terpisah.
   * **Observasi hasil** untuk memastikan apakah semua chunk diunggah dan digabungkan dengan benar.

# **ANALISIS HASIL EKSPERIMEN**

**Solusi 1: Meningkatkan Batas Maksimal dan Timeout pada Server**

* **Keberhasilan Upload**: File berukuran 90MB berhasil diunggah tanpa mengalami timeout setelah batas maksimal dan timeout ditingkatkan.
* **Konsumsi Sumber Daya**: Server menggunakan lebih banyak memori dan CPU saat mengunggah file besar dalam satu kali proses.
* **Ketahanan**: Jika koneksi terganggu atau terputus, seluruh file harus diunggah ulang, yang memakan waktu dan bandwidth.

Solusi ini efektif untuk file berukuran menengah dan kondisi jaringan yang stabil. Namun, konsumsi sumber daya yang tinggi dan ketergantungan pada kestabilan koneksi membuatnya kurang cocok untuk file yang sangat besar atau kondisi jaringan yang buruk.

**Solusi 2: Menggunakan Chunked Uploads**

* **Keberhasilan Upload**: File berhasil diunggah dan digabungkan dari beberapa chunk. Setiap chunk diunggah secara terpisah, dan server berhasil menyusun ulang file tanpa masalah.
* **Konsumsi Sumber Daya**: Server menggunakan sumber daya lebih efisien karena hanya memproses satu chunk dalam satu waktu.
* **Ketahanan**: Jika terjadi kegagalan atau koneksi terputus, hanya chunk yang gagal yang perlu diunggah ulang, sehingga lebih efisien dalam penggunaan bandwidth dan waktu.

Solusi ini lebih unggul untuk menangani file besar dan jaringan yang tidak stabil. Meskipun lebih kompleks untuk diimplementasikan, chunked uploads memberikan hasil yang lebih andal dan efisien, terutama dalam skenario dengan koneksi yang tidak stabil atau file yang sangat besar.