**PENGEMBANGAN WEB (TEORI)**

**LAPORAN EKSPERIMEN MENGENAI PROBLEM MASALAH PENANGANAN ERROR PADA FETCHING DATA SECARA PARALEL**

*Laporan ini disusun untuk memenuhi tugas 1 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

Septyana Agustina 221524058

Sarah 221524059

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](#_Toc176873917)

[**A.** **IDENTIFIKASI PROBLEM** 1](#_Toc176873918)

[**B.** **DESKRIPSI PROBLEM** 1](#_Toc176873919)

[**C.** **METODOLOGI EKSPERIMEN** 1](#_Toc176873920)

[**D.** **PELAKSANAAN EKSPERIMEN** 2](#_Toc176873921)

[**E.** **ANALISIS HASIL EKSPERIMEN** 3](#_Toc176873922)

[**F.** **KESIMPULAN** 3](#_Toc176873923)

Link chatGPT : https://chatgpt.com/share/5277fea4-2c90-492d-9cd1-9a70c18491bf

# **IDENTIFIKASI PROBLEM**

Penggunaan Promise.all() untuk melakukan fetching data secara paralel dapat menyebabkan kegagalan keseluruhan proses jika salah satu promise mengalami error, sehingga data dari promise yang berhasil tidak dapat diakses. Masalah ini dapat mengganggu pengalaman pengguna dan stabilitas aplikasi, terutama ketika ada kemungkinan kegagalan parsial.

# **DESKRIPSI PROBLEM**

* Dengan Promise.all(), jika salah satu promise gagal (rejected), seluruh Promise.all() akan gagal dan error akan dilemparkan. Hasil dari promise yang berhasil tidak dapat diakses.
* Ini berisiko menyebabkan kehilangan data yang berhasil diambil dan dapat membuat aplikasi kurang stabil dalam kondisi jaringan yang tidak dapat diandalkan atau ketika berurusan dengan API eksternal yang mungkin tidak selalu dapat diakses.
* Sebagai contoh, dalam aplikasi yang membutuhkan data dari beberapa sumber API, kegagalan satu API dapat mengakibatkan seluruh aplikasi gagal untuk menampilkan informasi yang valid.

# **METODOLOGI EKSPERIMEN**

1. **Simulasi Request API:** Buat tiga fungsi API fiktif menggunakan fetch(); dua diantaranya sukses dan satu sengaja dibuat gagal.
2. **Pendekatan 1:** Gunakan Promise.all() untuk menjalankan request secara paralel dan amati hasil ketika salah satu request gagal.
3. **Pendekatan 2:** Gunakan Promise.allSettled() untuk menangani setiap promise secara individual dan evaluasi bagaimana setiap hasil ditangani ketika ada kegagalan.

# **PELAKSANAAN EKSPERIMEN**

|  |
| --- |
| **Pendekatan 1: Menggunakan Promise.all()** |
|  |
| **Hasil:**     * Ketika fetchStocks() gagal, Promise.all() menghentikan seluruh proses dan tidak ada data yang ditampilkan. * Hanya pesan error "One of the requests failed: Stocks API failed" yang muncul. |

|  |
| --- |
| **Pendekatan 2: Menggunakan Promise.allSettled()** |
|  |
| **Hasil:**     * fetchWeather() dan fetchNews() berhasil dan hasilnya ditampilkan. * fetchStocks() gagal, tetapi hasil dari request yang berhasil tetap diproses. |

# **ANALISIS HASIL EKSPERIMEN**

|  |
| --- |
| Pendekatan 1 (Promise.all()):   * Kegagalan satu permintaan menyebabkan seluruh proses gagal, mengakibatkan hilangnya data dari permintaan yang berhasil. * Ini mengurangi ketahanan aplikasi terhadap kegagalan parsial dan bisa menyebabkan pengalaman pengguna yang buruk.   Pendekatan 2 (Promise.allSettled()):   * Menangani setiap promise secara independen, sehingga kegagalan satu promise tidak menghentikan seluruh proses. Data dari promise yang berhasil tetap tersedia. * Memberikan fleksibilitas dalam penanganan hasil dan error, meningkatkan ketahanan dan kontrol dalam aplikasi. |

# **KESIMPULAN**

**Promise.all()**:

* **Kelebihan:**
  + Sederhana dan langsung untuk penggunaan umum.
  + Efisien jika semua promise diharapkan berhasil.
* **Kekurangan:**
  + Seluruh proses gagal jika salah satu promise gagal, sehingga data dari promise yang berhasil tidak dapat diakses.
  + Tidak memungkinkan penanganan error secara individual.
* **Kapan Digunakan:**
  + Ideal untuk skenario di mana semua promise saling bergantung dan harus berhasil bersama-sama.

**Promise.allSettled()**:

* **Kelebihan:**
  + Menangani hasil dari setiap promise secara individu, baik yang berhasil maupun yang gagal.
  + Memastikan data dari promise yang berhasil tetap diproses meskipun ada kegagalan.
* **Kekurangan:**
  + Membutuhkan kode tambahan untuk memeriksa dan menangani hasil setiap promise.
  + Mungkin lebih kompleks untuk kasus di mana semua promise diharapkan sukses.
* **Kapan Digunakan:**
  + Cocok untuk skenario di mana ada kemungkinan kegagalan pada beberapa promise dan Anda ingin memastikan semua data yang berhasil diambil tetap tersedia dan diproses.

Pemilihan antara Promise.all() dan Promise.allSettled() bergantung pada kebutuhan spesifik aplikasi dan toleransi terhadap kegagalan. Jika aplikasi harus terus berfungsi meskipun beberapa request gagal, Promise.allSettled() memberikan solusi yang lebih fleksibel dan tahan banting.