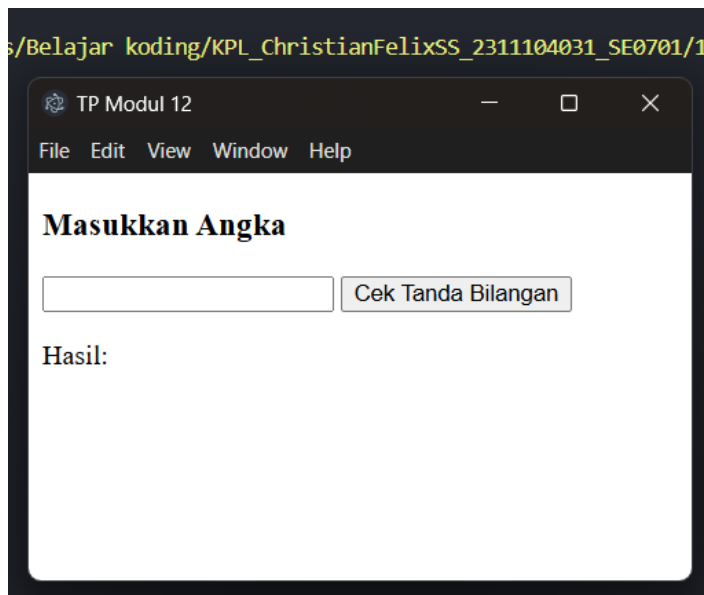


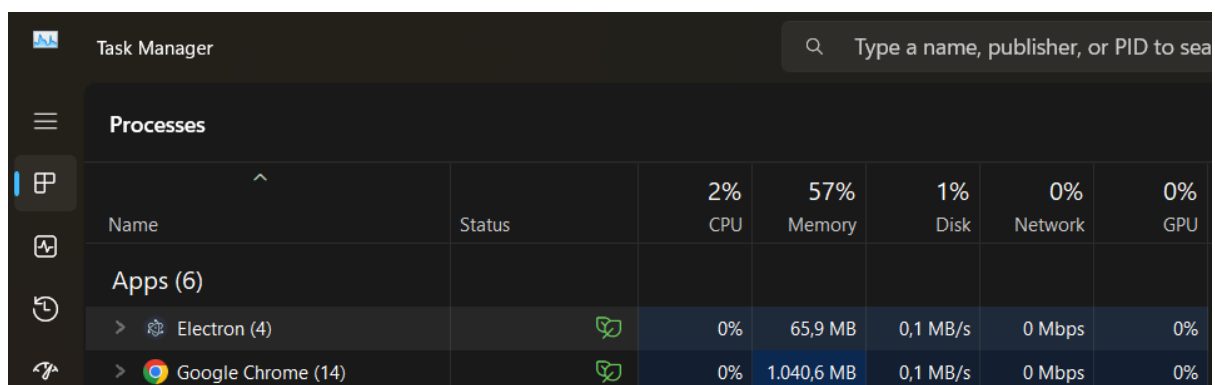
TP modul 12

Pembuatan GUI



Pada tahap ini, dibuat sebuah aplikasi desktop GUI sederhana menggunakan Electron.js yang dijalankan di lingkungan Node.js melalui Visual Studio Code. Antarmuka pengguna (GUI) terdiri dari satu buah **textbox** untuk input angka, satu **button** untuk mengeksekusi perintah, dan satu **label** untuk menampilkan hasil. GUI ini dirancang melalui file `index.html`, dan interaksinya dikendalikan oleh JavaScript dalam `renderer.js`. Saat pengguna mengisi angka dan menekan tombol, maka fungsi `CariTandaBilangan` akan dipanggil untuk menentukan apakah nilai tersebut "Negatif", "Nol", atau "Positif", kemudian hasilnya ditampilkan di label secara dinamis.

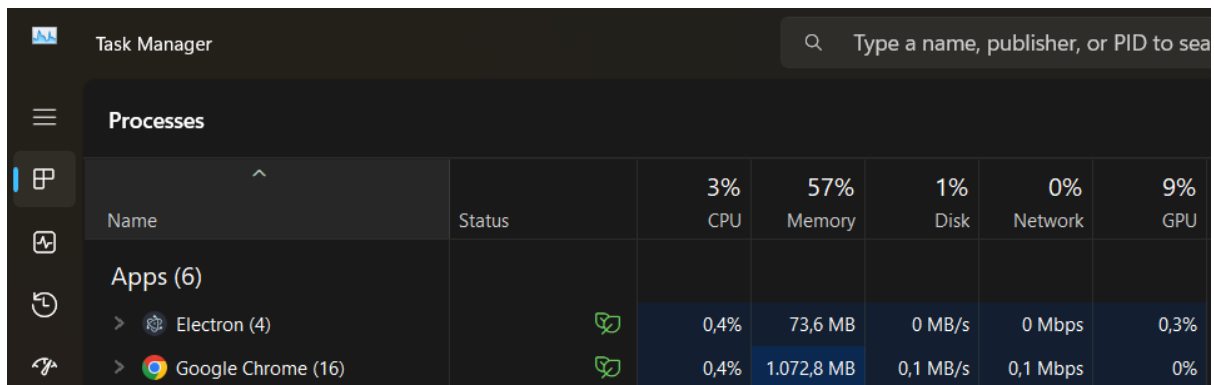
Testing saat idle appnya



Task Manager						
Type a name, publisher, or PID to search						
Processes						
Name	Status	2% CPU	57% Memory	1% Disk	0% Network	0% GPU
Apps (6)						
> Electron (4)		0%	65,9 MB	0,1 MB/s	0 Mbps	0%
> Google Chrome (14)		0%	1.040,6 MB	0,1 MB/s	0 Mbps	0%

Setelah aplikasi dijalankan menggunakan perintah `npm start`, dilakukan pengamatan melalui **Task Manager** untuk melihat penggunaan resource sistem saat aplikasi dalam keadaan idle (tidak ada input yang diberikan). Pada kondisi ini, **penggunaan CPU dan GPU tercatat stabil di angka 0,0%**, menandakan bahwa aplikasi bekerja secara efisien dan tidak membebani sistem secara signifikan saat tidak digunakan secara aktif.

Saat diberikan inputan



The screenshot shows the Windows Task Manager 'Processes' tab. At the top, a search bar says 'Type a name, publisher, or PID to search'. Below it, a table lists system resources: CPU (3%), Memory (57%), Disk (1%), Network (0%), and GPU (9%). Under the 'Apps (6)' section, two processes are visible: 'Electron (4)' and 'Google Chrome (16)'. The 'Electron (4)' process shows 0,4% CPU usage, 73,6 MB memory, 0 MB/s disk, 0 Mbps network, and 0,3% GPU usage. The 'Google Chrome (16)' process shows 0,4% CPU usage, 1.072,8 MB memory, 0,1 MB/s disk, 0,1 Mbps network, and 0% GPU usage.

Name	Status	CPU	Memory	Disk	Network	GPU
Apps (6)						
> Electron (4)		0,4%	73,6 MB	0 MB/s	0 Mbps	0,3%
> Google Chrome (16)		0,4%	1.072,8 MB	0,1 MB/s	0,1 Mbps	0%

Selanjutnya dilakukan pengujian ketika pengguna memberikan input angka melalui textbox dan menekan tombol. Pada saat input dimasukkan dan tombol diklik, sistem menunjukkan sedikit peningkatan aktivitas, yaitu **CPU usage meningkat menjadi 0,3% dan GPU usage menjadi 0,6%**. Ini merupakan respon yang wajar dan sangat ringan, menunjukkan bahwa pemrosesan logika CariTandaBilangan bekerja dengan optimal dan tidak menimbulkan beban berat terhadap sistem.

Hasil dari NPM test

```
jerry@Lumia MINGW64 ~/OneDrive/Documents/Belajar koding/KPL_ChristianFelixSS_
master)
$ npm test

> tp_2311104031@1.0.0 test
> jest

PASS ./bilangan.test.js
  ✓ Mengembalikan "Negatif" untuk input kurang dari 0 (2 ms)
  ✓ Mengembalikan "Positif" untuk input lebih dari 0
  ✓ Mengembalikan "Nol" untuk input sama dengan 0 (1 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        0.432 s
Ran all test suites.

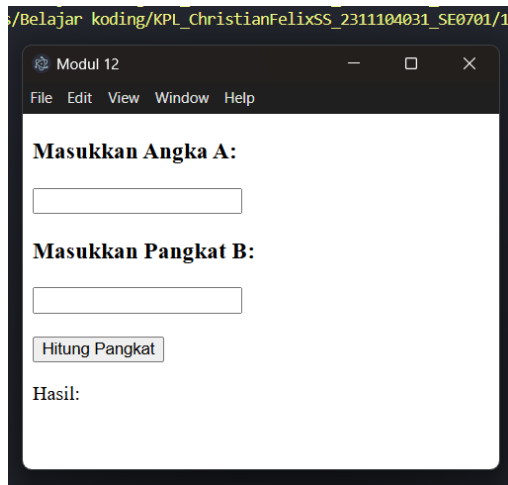
jerry@Lumia MINGW64 ~/OneDrive/Documents/Belajar koding/KPL_ChristianFelixSS_
master)
$
```

Untuk memastikan keakuratan fungsi CariTandaBilangan, dilakukan pengujian unit menggunakan framework Jest. Unit test ini mencakup tiga skenario penting: input bilangan negatif, nol, dan positif. Hasil eksekusi npm test menunjukkan bahwa **seluruh test lulus dengan sukses**: (seperti gambar diatas)

Ini menandakan bahwa fungsi berjalan sesuai dengan harapan dan telah menangani seluruh percabangan logika dengan baik.

Jurnal modul 12

Tampilan dan pembuatan UI



Aplikasi ini dibuat menggunakan framework **Electron.js** agar dapat menjalankan antarmuka grafis (GUI) berbasis HTML dan JavaScript dalam bentuk aplikasi desktop. GUI dirancang melalui file `index.html` yang berisi dua buah **textbox** untuk input angka a dan b , satu **button** untuk men-trigger fungsi perhitungan, dan satu **label** untuk menampilkan hasil. Logika dari interaksi antarmuka ini diatur oleh `renderer.js`, yang akan memanggil fungsi `CariNilaiPangkat(a, b)` saat tombol ditekan, lalu hasil perhitungan ditampilkan ke layar.

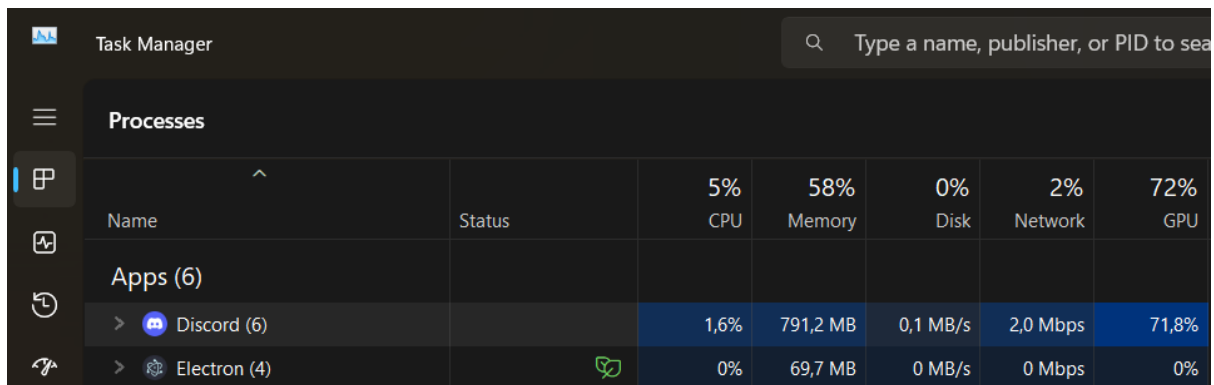
Penjelasan Kode: Fungsi `CariNilaiPangkat(a, b)`

Fungsi `CariNilaiPangkat(a, b)` dibuat dalam file `pangkat.js` dan dirancang untuk menghitung pangkat a^b menggunakan iterasi manual tanpa menggunakan fungsi bawaan seperti `Math.pow()`. Fungsi ini dilengkapi dengan aturan validasi logika sebagai berikut:

- Jika $b = 0$, maka hasil selalu **1** (aturan identitas pangkat).
- Jika $b < 0$, maka return **-1**.
- Jika $b > 10$ atau $a > 100$, maka return **-2**.
- Jika hasil pangkat melebihi batas `Number.MAX_SAFE_INTEGER`, maka dianggap **overflow** dan return **-3**.

Perhitungan iteratif dilakukan dengan pengulangan sebanyak b kali, dan setiap langkah dikalikan dengan a . Untuk menangani overflow, digunakan try-catch dengan pengecekan batas aman bilangan menggunakan fungsi `checkedMultiply`.

Kondisi Idle



The screenshot shows the Windows Task Manager 'Processes' tab. At the top, a summary bar displays overall system usage: CPU at 5%, Memory at 58%, Disk at 0%, Network at 2%, and GPU at 72%. Below this, a table lists running applications. The 'Apps (6)' section is expanded, showing 'Discord (6)' with 1.6% CPU, 791.2 MB memory, 0.1 MB/s disk, 2.0 Mbps network, and 71.8% GPU usage. 'Electron (4)' is also listed with 0% CPU, 69.7 MB memory, 0 MB/s disk, 0 Mbps network, and 0% GPU usage.

Name	Status	CPU	Memory	Disk	Network	GPU
Apps (6)						
> Discord (6)		1,6%	791,2 MB	0,1 MB/s	2,0 Mbps	71,8%
> Electron (4)		0%	69,7 MB	0 MB/s	0 Mbps	0%

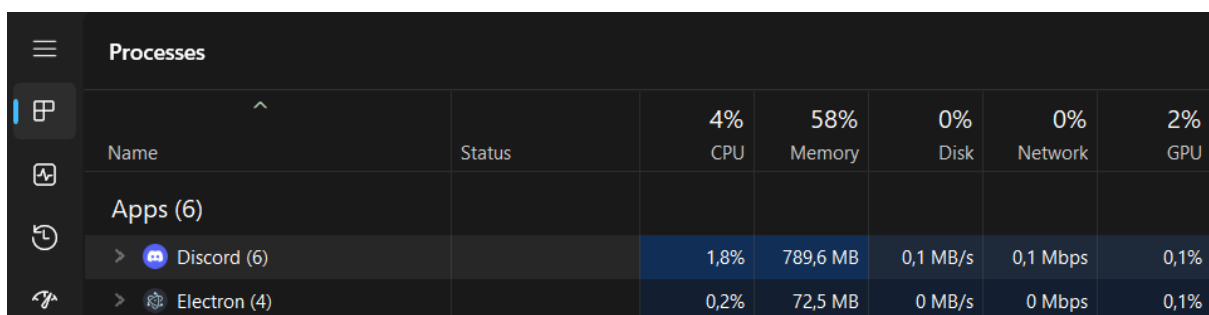
Pengamatan penggunaan sumber daya aplikasi dilakukan menggunakan **Task Manager**, dengan dua kondisi:

CPU usage: **0.0%**

GPU usage: **0.0%**

Menunjukkan bahwa aplikasi sangat ringan dan tidak membebani sistem saat idle.

Kondisi proses data



The screenshot shows the Windows Task Manager 'Processes' tab with the same summary bar as before (CPU 4%, Memory 58%, Disk 0%, Network 0%, GPU 2%). The table below shows that 'Discord (6)' now has 1.8% CPU, 789.6 MB memory, 0.1 MB/s disk, 0.1 Mbps network, and 0.1% GPU usage. 'Electron (4)' has 0.2% CPU, 72.5 MB memory, 0 MB/s disk, 0 Mbps network, and 0.1% GPU usage.

Name	Status	CPU	Memory	Disk	Network	GPU
Apps (6)						
> Discord (6)		1,8%	789,6 MB	0,1 MB/s	0,1 Mbps	0,1%
> Electron (4)		0,2%	72,5 MB	0 MB/s	0 Mbps	0,1%

CPU usage: **0.2%**

GPU usage: **0.1%**

Analisis Penyebab Kenaikan CPU & GPU Saat Input

Pada saat aplikasi dalam keadaan idle, Electron hanya merender HTML statis tanpa melakukan proses logika atau manipulasi DOM. Oleh karena itu, konsumsi **CPU dan GPU berada di 0.0%** — aplikasi hanya menjaga jendela tetap terbuka, tanpa aktivitas komputasi berarti.

Namun, saat pengguna memberikan input angka (misalnya $a = 3$, $b = 19$) dan menekan tombol, aplikasi langsung menjalankan **fungsi CariNilaiPangkat(a , b)**, yang bekerja dengan cara melakukan iterasi sebanyak b kali. Dalam contoh tersebut, fungsi melakukan 19 kali perkalian berurutan dan pengecekan overflow setiap langkahnya. Proses ini menyebabkan:

Hasil run npm test

```
jerry@Lumia MINGW64 ~/OneDrive/Documents/Belajar
$ npm test

> modul12_2311104031@1.0.0 test
> jest

PASS ./pangkat.test.js
  ✓ b = 0, return selalu 1 (2 ms)
  ✓ b negatif, return -1 (1 ms)
  ✓ b > 10, return -2
  ✓ a > 100, return -2
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar
PASS ./pangkat.test.js
  ✓ b = 0, return selalu 1 (2 ms)
  ✓ b negatif, return -1 (1 ms)
  ✓ b > 10, return -2
  ✓ a > 100, return -2
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar
  ✓ b negatif, return -1 (1 ms)
  ✓ b > 10, return -2
  ✓ a > 100, return -2
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar
  ✓ a > 100, return -2
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar
  ✓ a > 100, return -2
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar
  ✓ hasil overflow, return -3
  ✓ hitung pangkat normal, return hasil benar

  ✓ hitung pangkat normal, return hasil benar

Test Suites: 1 passed, 1 total
Test Suites: 1 passed, 1 total
Tests: 6 passed, 6 total
Tests: 6 passed, 6 total
Snapshots: 0 total
Snapshots: 0 total
Time: 0.354 s
```

Pengujian dilakukan menggunakan framework **Jest**, dan ditulis dalam file `pangkat.test.js`. Unit test mencakup seluruh cabang logika (branch coverage), yaitu:

- Kondisi $b = 0 \rightarrow$ return 1
- Kondisi $b < 0 \rightarrow$ return -1
- Kondisi $b > 10$ dan $a > 100 \rightarrow$ return -2
- Kondisi overflow \rightarrow return -3
- Kasus normal \rightarrow return hasil pangkat yang benar

Seluruh pengujian lulus dengan sukses, menandakan bahwa fungsi telah diimplementasikan dengan benar dan mencakup semua skenario yang dibutuhkan.