

# Mobile Web Modern

Panduan pengembangan mobile web fokus pada pengguna



Yohan Totting

@tyohan

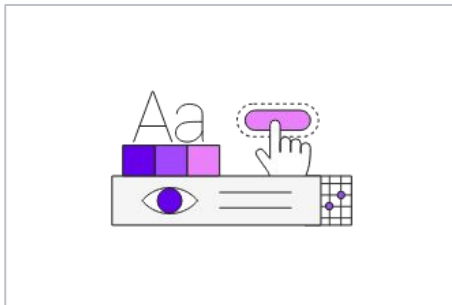
## Yang Perlu Disiapkan

1. Laptop yang sudah terpasang aplikasi untuk pengembangan web
2. Chrome browser terbaru
3. Perangkat Android beserta kabel charger

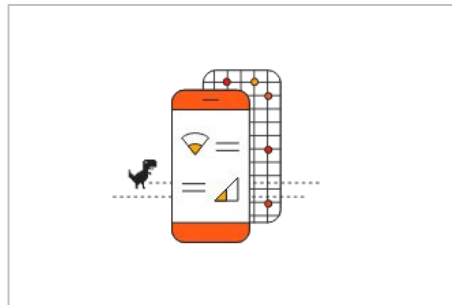
# Topik berdasarkan web.dev



Cepat



Aksesibilitas



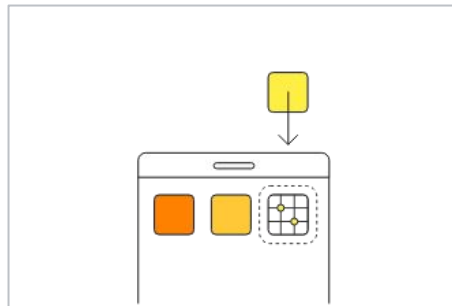
Nyaman di jaringan apapun



Bisa ditemukan



Aman



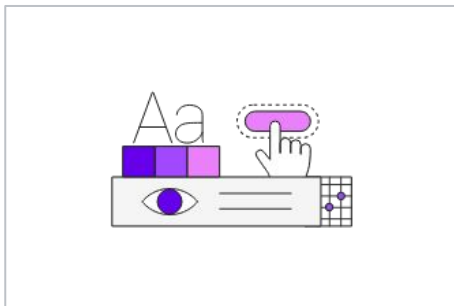
Bisa di-install

# Topik

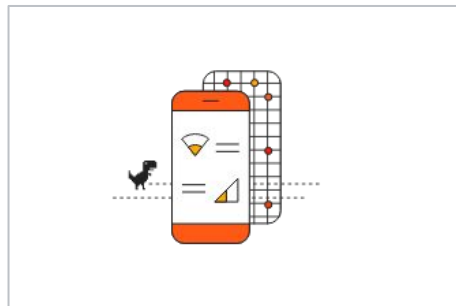
Proprietary + Confidential



Cepat



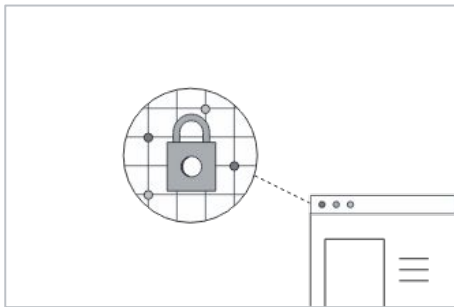
Aksesibilitas



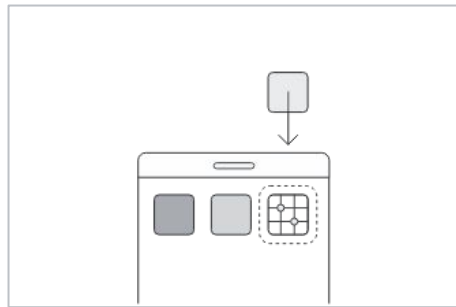
Nyaman di jaringan  
apapun



Bisa ditemukan



Aman



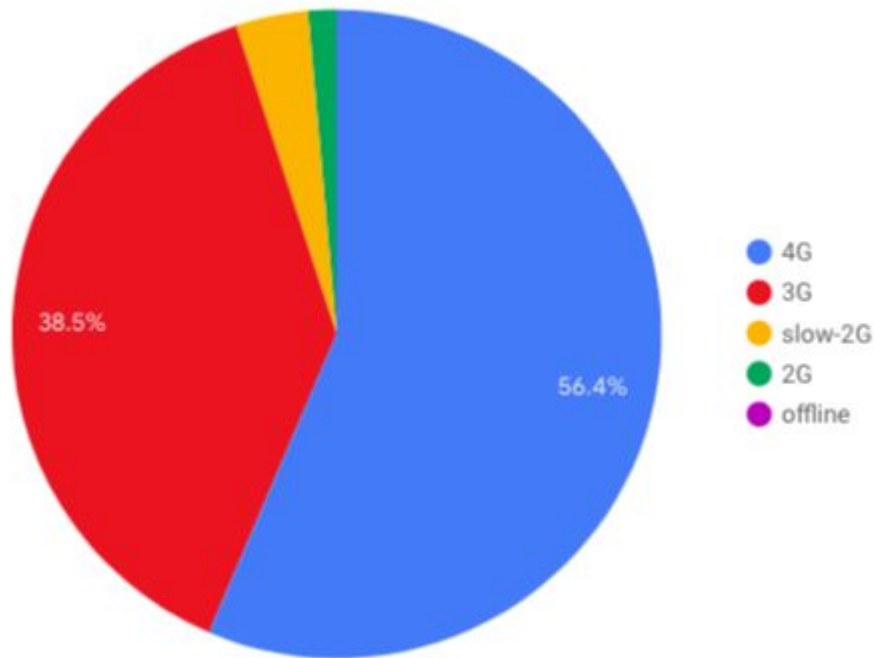
Bisa di-install

# Cepat

Halaman dimuat dalam 3 detik



# Koneksi efektif di Indonesia berdasarkan CrUX



# Aktifitas: Chrome Remote Debugging

Untuk mengetahui bagaimana sebenarnya sebuah website bisa dimuat di perangkat mobile dan jaringan 3G, pastikan menguji website di perangkat mobile sebenarnya dengan jaringan yang selular.

1. Aktifkan remote debugging dengan perangkat mobile
2. Coba remote debugging sebuah website dengan Chrome DevTools

# Aktifitas: Mengukur dan Menganalisa Performance

1. Menganalisa apa yang perlu dioptimalisasi
  - a. Intro Chrome DevTools
    - i. Network panel untuk memeriksa konten yang dimuat
    - ii. Performance panel untuk memeriksa bagaimana resource dieksekusi
    - iii. Audit panel untuk mengukur dengan Lighthouse
    - iv. Application panel untuk debugging PWA
  - b. Mengukur dengan Lighthouse
    - i. Dengan Chrome Devtools audit panel
    - ii. Dengan Lighthouse CLI
  - c. Mengukur dengan WebPageTest  
<https://www.webpagetest.org/easy.php>
    - i. (contoh: [wwwid.org](http://wwwid.org))



Chrome

Remote

Debugging

**GOOGLE CHROME  
REMOTE DEBUGGING  
ON ANDROID**



# Aktifitas: Optimasi Performance

Gunakan panduan pada Lighthouse bagian performance apa yang perlu dioptimasi. Biasanya masalah performance paling banyak disebabkan oleh kode dan gambar yang tidak optimal.

1. Optimasi kode

<https://web.dev/codelab-preload-critical-assets/>

- a. Async & Defer Scripts
- b. Preload aset penting

2. Optimasi gambar

<https://web.dev/codelab-use-lazysizes-to-lazyload-images/>

- a. Format, ukuran, dan kompresi yang tepat
- b. Lazy load

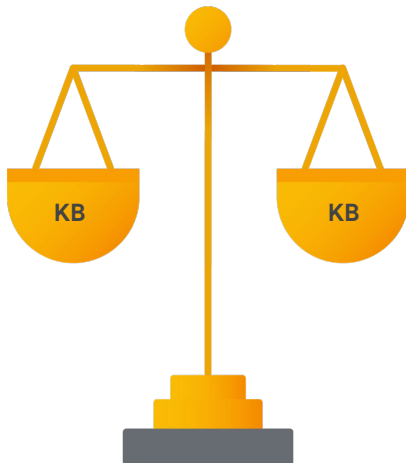
# Performance Budget

1. Performance budget calculator
2. Lightwallet



**Time**

< 2 second TTI



**Resources**

< 150KB JS

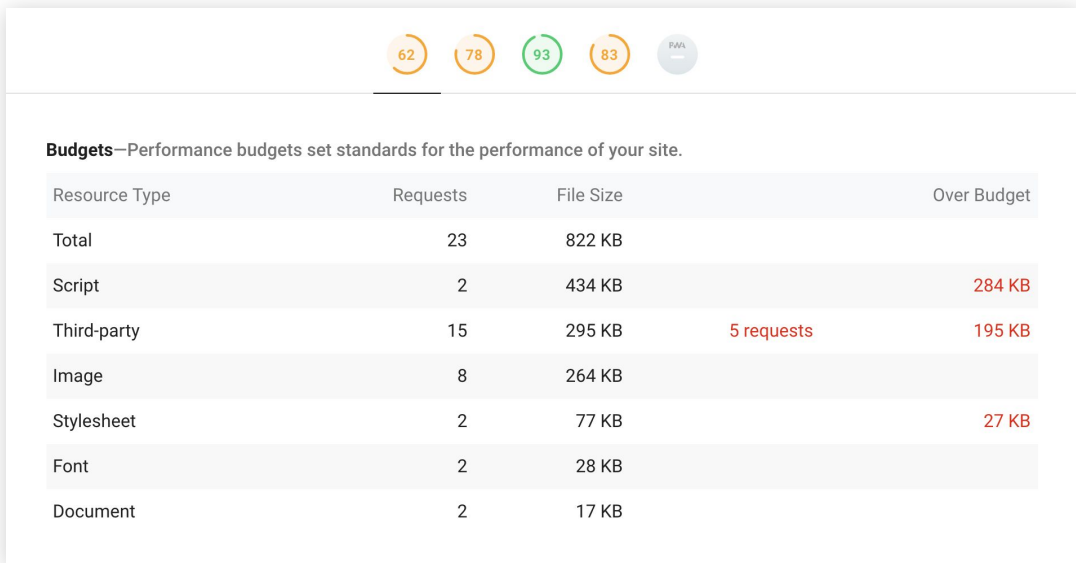


**Lighthouse**

90+ Perf Score



## Performance budget tooling for Lighthouse



1. Add a budget.json file

2. Run Lighthouse **5.0+** from the **CLI**

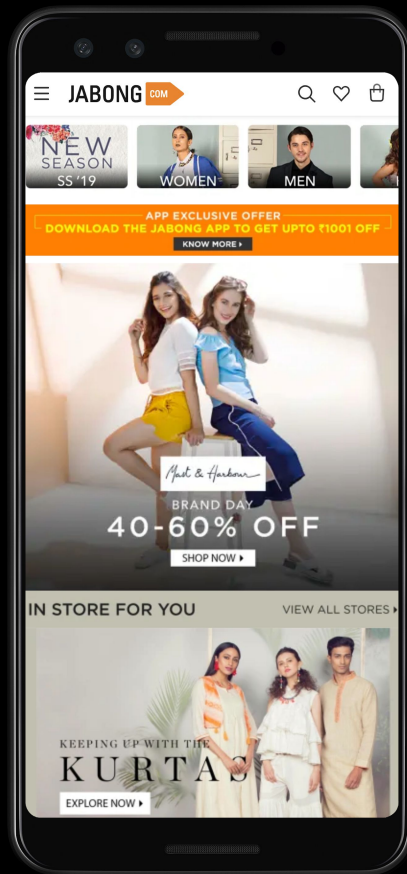
```
$ lighthouse https://example.com --budgetPath ./budget.json
```

# budget.json

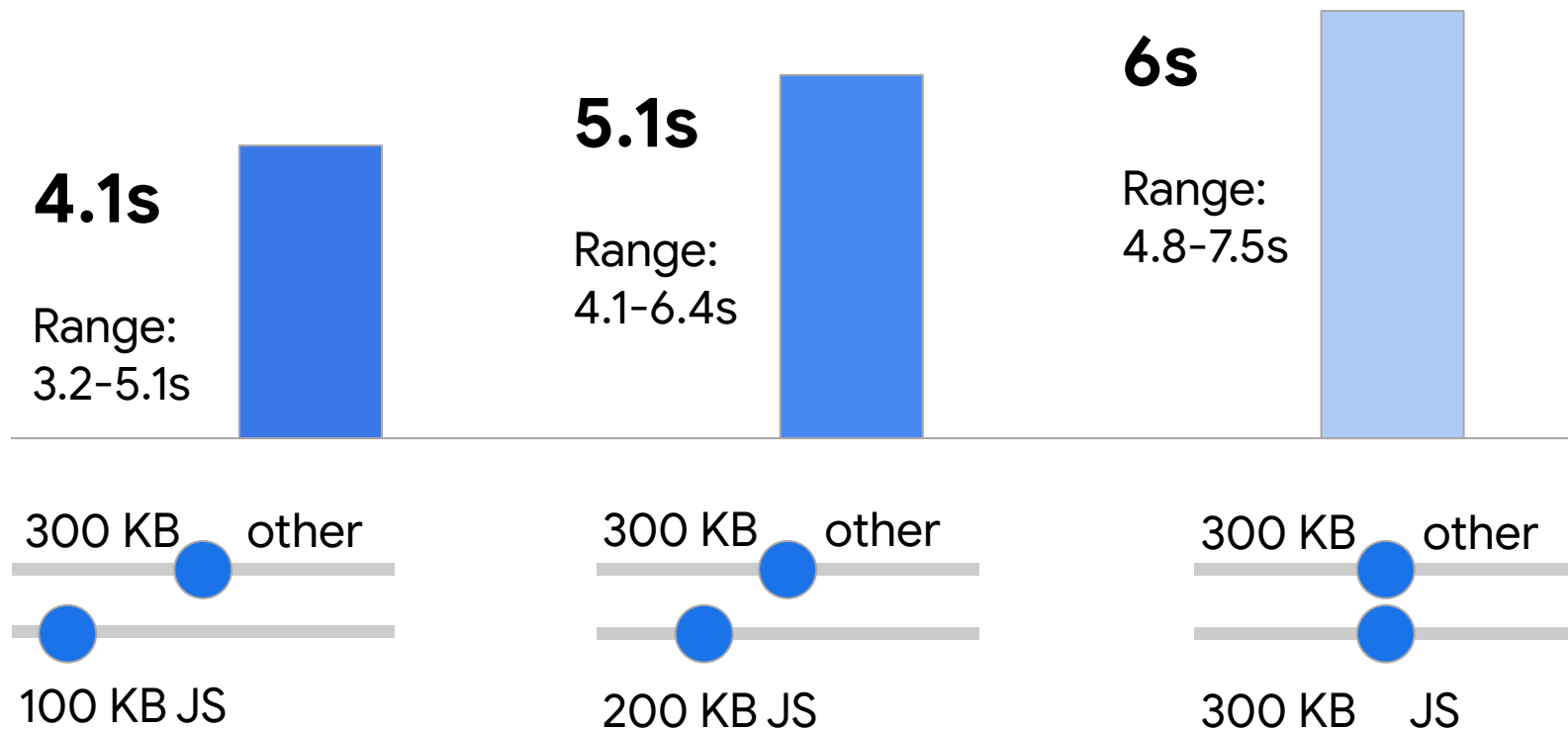
## Budget types:

- Resource Sizes
- Resource Counts

```
[{
  "resourceSizes": [
    {
      "resourceType": "image",
      "budget": 250
    },
    {
      "resourceType": "script",
      "budget": 125
    },
    {
      "resourceType": "font",
      "budget": 30
    },
    {
      "resourceType": "document",
      "budget": 25
    },
    {
      "resourceType": "stylesheet",
      "budget": 5
    },
    {
      "resourceType": "third-party",
      "budget": 400
    }
  ]
}]
```



**JABONG** COM



[bit.ly/perf-budget-calculator](https://bit.ly/perf-budget-calculator)



# Aktifitas: Menerapkan performance budget

1. Tentukan performance budget menggunakan [Performance budget calculator](#)
2. Ukur website dengan Lighthouse + performance budget
3. Lakukan optimasi ulang performance agar bisa sesuai dengan budget yang ditetapkan.

# Get fast

# Stay Fast

1. Lazy-loading
2. Responsive Images
3. Image CDNs
4. Defer Third-party JS
5. Remove costly libraries
6. Code-Splitting
7. Avoid FOIT

8. Prefetch
9. Preconnect
10. Preload
11. Critical CSS
12. Brotli
13. Adaptive Serving

Performance Budgets



**LightWallet**

# Aksesibilitas

Website yang bisa diakses oleh semua

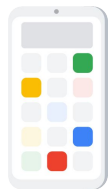


# Apa itu aksesibilitas



## Visi

Kemudahan akses untuk pengguna dengan keterbatasan kemampuan melihat seperti buta warna, buta partial, rabun, dan lainnya.



## Motorik

Kemudahan akses untuk interaktivitas dengan medium yang terbatas seperti keyboard, mouse, layar sentuh, dan lainnya.



## Pendengaran

Kemudahan akses untuk pendengaran seperti keterbatasan kemampuan pendengaran ataupun kondisi lingkungan yang membatasi suara.



## Kognitif

Kenyamanan akses dengan mengurangi distraksi yang membuat tidak nyaman seperti animasi, layar yang berkedip, atau hal-hal yang bisa mendistraksi fokus pengguna

# Website dengan aksesibilitas yang baik

## Navigasi dengan apa saja

Tidak terbatas oleh layar sentuh, keypad, keyboard, ataupun mouse.

## Web yang semantic

Prioritas dan role sebuah elemen bisa dipahami dengan baik screen reader, bot, atau medium lainnya.

# Hati-hati dengan perubahan layout element bisa merusak navigasi



```
<button>Kiwi</button>  
<button>Peach</button>  
<button>Coconut</button>
```

```
<button style="float: right">Kiwi</button>  
<button>Peach</button>  
<button>Coconut</button>
```

# Navigasi halaman dengan keyboard atau keypad

Gunakan

```
<button>Klik ini</button>
```

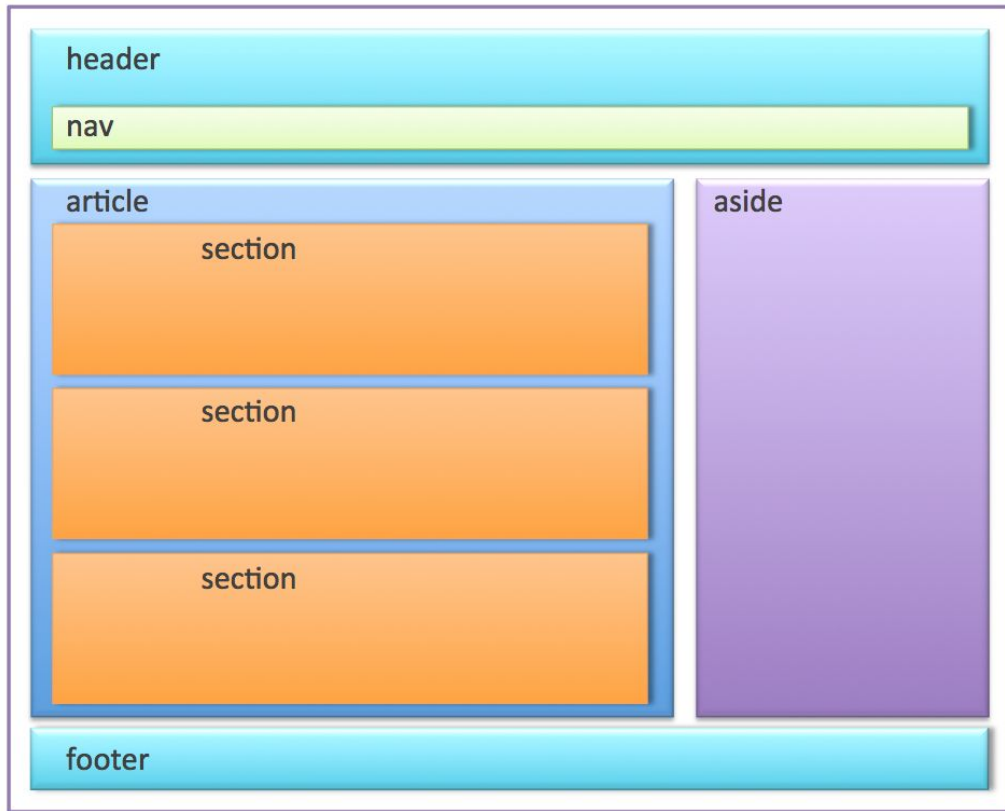
Bukan

```
<a href="#ini" onclick="ini()">Klik ini</a>
```

apalagi

```
<div onclick="ini()">Klik ini</div>
```

# Semantic layout



```
<header>  
<nav>  
    <a  
href="menu.html">menu1</a>  
</nav>  
</header>
```

```
<ul>
```

```
<li>  
</ul>
```



# Aktifitas: Mengikuti panduan Lighthouse untuk accessibility

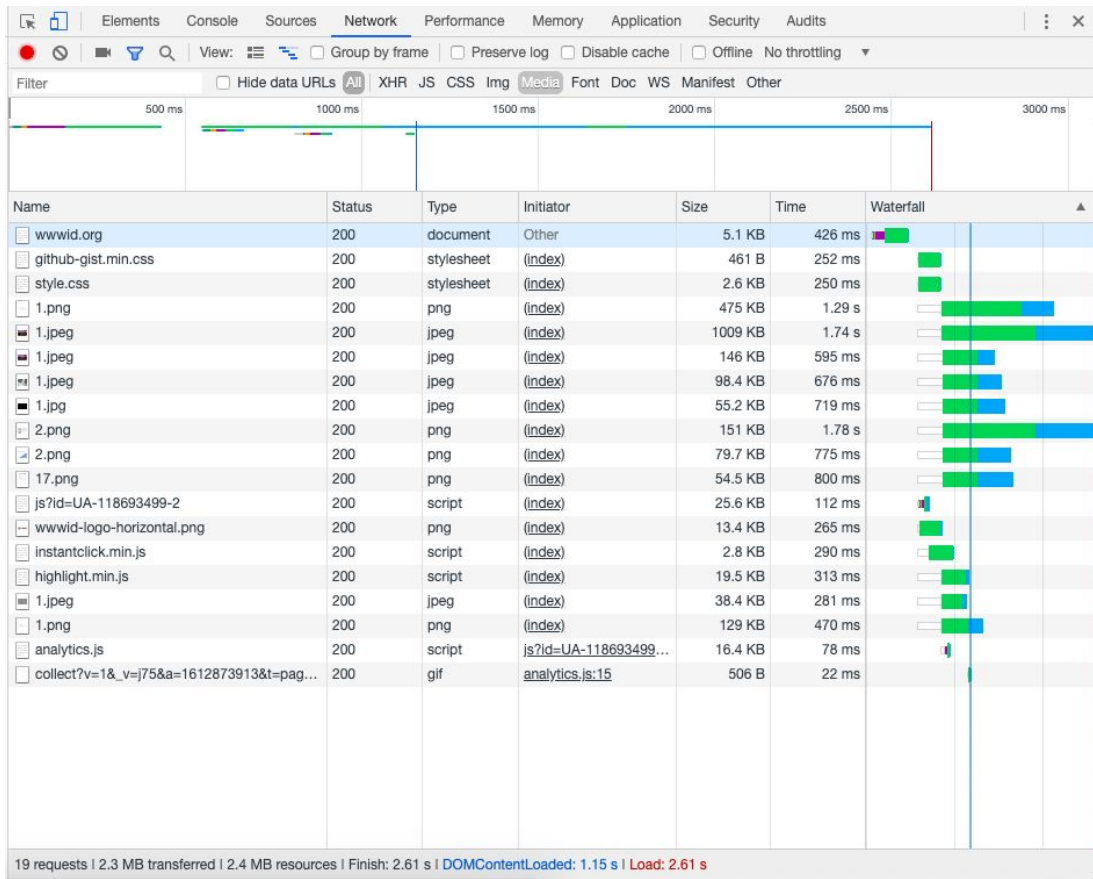
1. Uji website dengan Lighthouse
2. Cek hasil pada bagian accessibility
3. Ikuti panduan untuk hingga skor Lighthouse untuk accessibility bisa  $> 80$ .

# Nyaman di jaringan apapun

Baik offline maupun 2G



# Network Panel - Chrome DevTools



# Checklist

- 1 Pastikan menggunakan HTTP cache headers
- 2 Utamakan kondisi offline
- 3 Merancang strategi offline sesuai dengan fungsi website
- 4 Gunakan service worker dan Cache API untuk kondisi jaringan yang tidak optimal

# HTTP Cache

# Request Header

```
GET /home.html HTTP/1.1
Host: developer.mozilla.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:50.0)
Gecko/20100101 Firefox/50.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://developer.mozilla.org/testpage.html
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-Modified-Since: Mon, 18 Jul 2016 02:36:04 GMT
If-None-Match: "c561c68d0ba92bb8b0fff2a9199f722e3a621a"
Cache-Control: max-age=0
```

# Response Header

```
200 OK
Access-Control-Allow-Origin: *
Connection: Keep-Alive
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Date: Mon, 18 Jul 2016 16:06:00 GMT
Etag: "c561c68d0ba92bb8b0f612a9199f722e3a621a"
Keep-Alive: timeout=5, max=997
Last-Modified: Mon, 18 Jul 2016 02:36:04 GMT
Server: Apache
Set-Cookie: mykey=myvalue; expires=Mon, 17-Jul-2017 16:06:00 GMT;
Max-Age=31449600; Path=/; secure
Transfer-Encoding: chunked
Vary: Cookie, Accept-Encoding
```

# Response Header - Cache Control

Enable Cache untuk aset statis seperti gambar, CSS, atau JavaScript

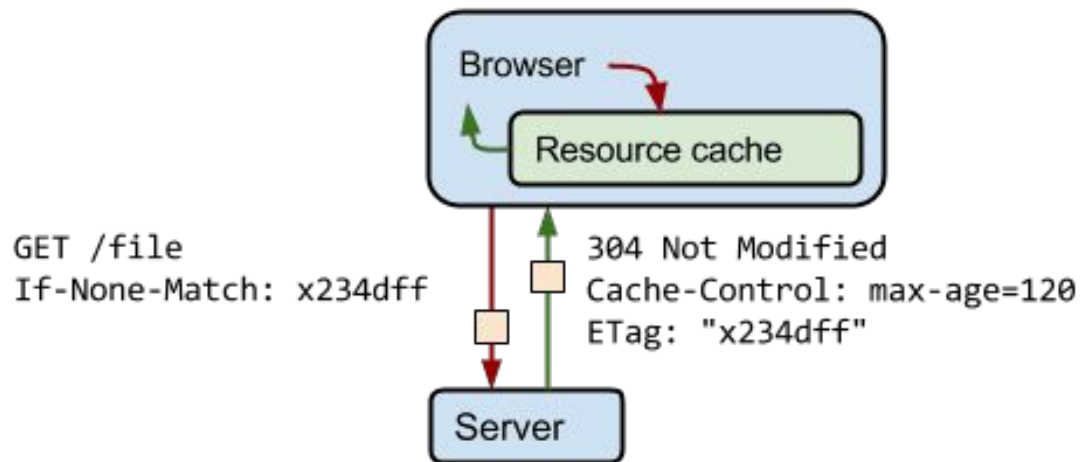
```
Cache-Control: max-age=31536000
```

Disable cache untuk API seperti JSON

```
Cache-Control: no-cache
```



# Response Header - Cache Control



# Cek Cache Headers di Chrome DevTool - Network

The screenshot shows the Chrome DevTools Network tab. The left sidebar lists 19 requests, with the first one, `wwwid.org`, selected. The main panel displays the response headers for this request. The headers are categorized into General and Response Headers. The General section shows the Request URL, Method, Status Code, Remote Address, and Referrer Policy. The Response Headers section lists various cache-related headers, including `accept-ranges`, `access-control-allow-origin`, `age`, `cache-control`, `content-encoding`, `content-length`, `content-type`, `date`, `etag`, `expires`, `last-modified`, `server`, `status`, `strict-transport-security`, `vary`, `via`, `x-cache`, and `x-cache-hits`.

**General**

- Request URL: `https://wwwid.org/`
- Request Method: `GET`
- Status Code: `200`
- Remote Address: `185.199.109.153:443`
- Referrer Policy: `no-referrer-when-downgrade`

**Response Headers**

- `accept-ranges: bytes`
- `access-control-allow-origin: *`
- `age: 0`
- `cache-control: max-age=600`
- `content-encoding: gzip`
- `content-length: 4801`
- `content-type: text/html; charset=utf-8`
- `date: Fri, 24 May 2019 05:03:21 GMT`
- `etag: W/"5ce762f1-3b42"`
- `expires: Fri, 24 May 2019 03:32:49 GMT`
- `last-modified: Fri, 24 May 2019 03:20:17 GMT`
- `server: GitHub.com`
- `status: 200`
- `strict-transport-security: max-age=31556952`
- `vary: Accept-Encoding`
- `via: 1.1 varnish`
- `x-cache: HIT`
- `x-cache-hits: 1`

19 requests | 2.3 MB transferred | ...

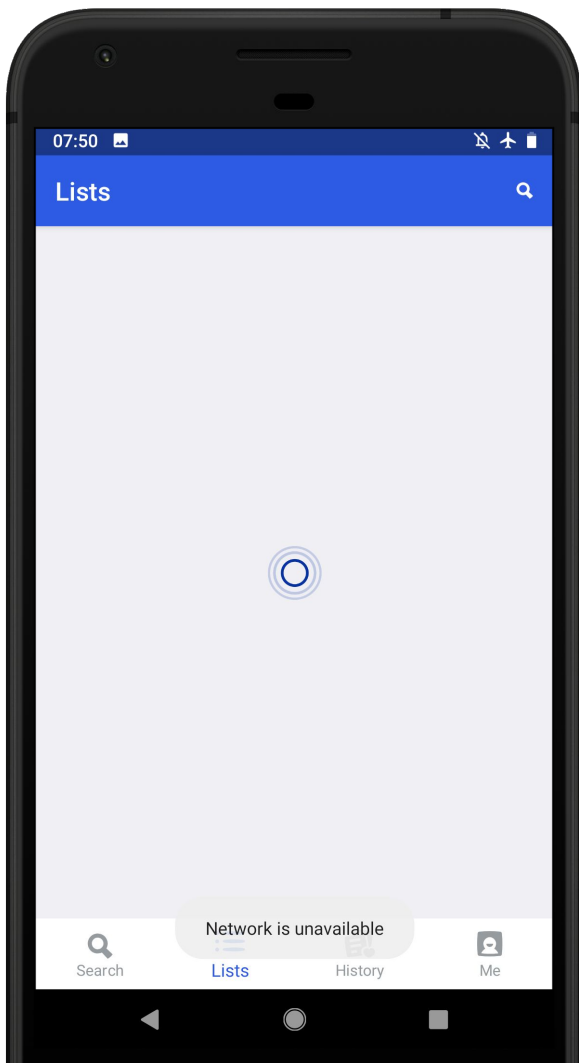
# Aktifitas: Pastikan HTTP Cache sudah diset untuk aset statis

**Diagnostics** — More information about the performance of your application.

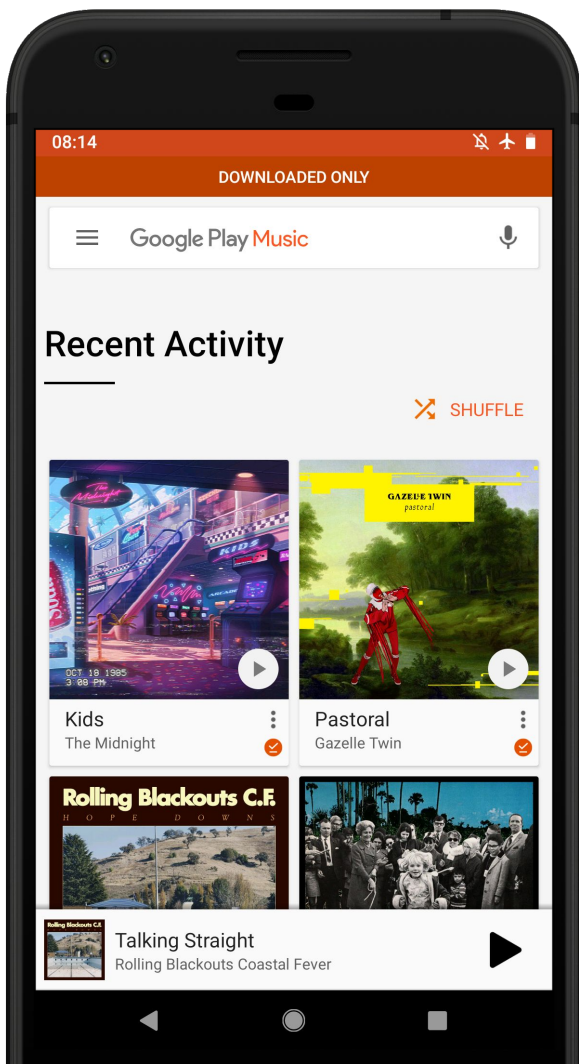
▲ Serve static assets with an efficient cache policy — 17 resources found ^

1. Cek hasil Lighthouse pada bagian performance
2. Pastikan aset statis sudah memiliki header response HTTP Cache yang tepat.

# Moda offline



# Pilihan 1: Tunjukkan halaman offline



Pilihan 2:  
Bila offline,  
tunjukkan konten  
yang sudah bisa  
diakses offline



Level 3:  
Langsung  
tunjukkan konten  
sebelumnya  
sambil update  
konten terbaru.

# Pemasangan service worker

- 1 Register service worker
- 2 Precache aset statis pada install event
- 3 Terapkan cache strategy pada fetch event

<https://web.dev/codelab-service-workers>



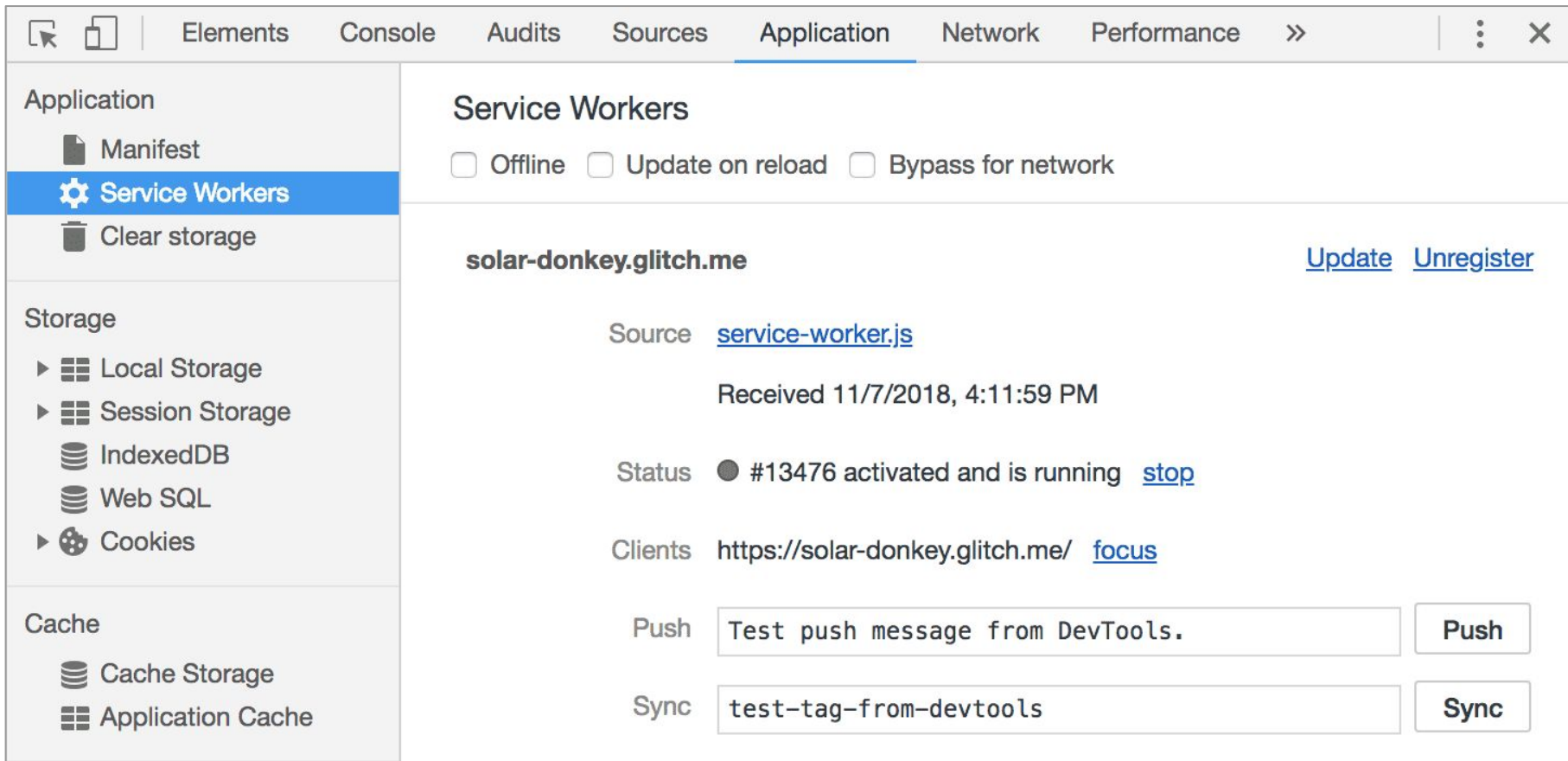
# Precache aset statis

```
self.addEventListener('install', function(event) {  
  event.waitUntil(  
    caches.open('static-assets').then(function(cache) {  
      return cache.addAll([  
        '/css/whatever-v3.css',  
        '/css/imgs/sprites-v6.png',  
        '/css/fonts/whatever-v8.woff',  
        '/js/all-min-v4.js',  
        '/offline.html' // cache untuk offline page  
      ]);  
    })  
  );  
});
```

# Handle fetch untuk offline page atau aset lainnya

```
this.addEventListener('fetch', event => {  
  // Untuk request halaman HTML kita akan fallback ke halaman offline  
  if (event.request.mode === 'navigate' || (event.request.method === 'GET' &&  
event.request.headers.get('accept').includes('text/html')))) {  
    event.respondWith(  
      fetch(event.request.url).catch(error => {  
        // Return the offline page  
        return caches.match('/offline.html');  
      })  
    );  
  }  
  else{  
    // Untuk aset lainnya ambil dari cache bila tersedia  
    event.respondWith(caches.match(event.request)  
      .then(function (response) {  
        return response || fetch(event.request);  
      })  
    );  
  }  
});
```

# Debugging service worker dengan Chrome Dev Tool



The screenshot shows the Chrome DevTools interface with the 'Application' tab selected. The left sidebar contains the 'Application' panel with 'Service Workers' highlighted. The main area displays details for a service worker from 'solar-donkey.glitch.me'. It includes controls for 'Offline', 'Update on reload', and 'Bypass for network'. The source is 'service-worker.js' and it was received on 11/7/2018 at 4:11:59 PM. The status is '#13476 activated and is running' with a 'stop' link. The client is 'https://solar-donkey.glitch.me/' with a 'focus' link. At the bottom, there are input fields for a 'Push' message ('Test push message from DevTools.') and a 'Sync' tag ('test-tag-from-devtools'), each with a corresponding button.

Application

- Manifest
- Service Workers**
- Clear storage

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

Cache

- Cache Storage
- Application Cache

## Service Workers

☐ Offline ☐ Update on reload ☐ Bypass for network

**solar-donkey.glitch.me** [Update](#) [Unregister](#)

Source [service-worker.js](#)

Received 11/7/2018, 4:11:59 PM

Status ● #13476 activated and is running [stop](#)

Clients <https://solar-donkey.glitch.me/> [focus](#)

Push

Sync

# Aktifitas: Pastikan service worker terpasang

1. Pasang service worker pada website
2. Pastikan service worker terpasang dan berjalan dengan baik melalui Chrome DevTools application panel
3. Precache aset statis dengan service worker di install event
4. Cache konten secara dinamis di fetch event
5. Pasang offline mode pada fetch event
6. Uji dengan moda offline menggunakan network throttle switch di Chrome DevTools

Terima kasih