

DIGITAL TALENT SCHOLARSHIP 2019



Sesi 12

Data Crawling

Big Data Analytics

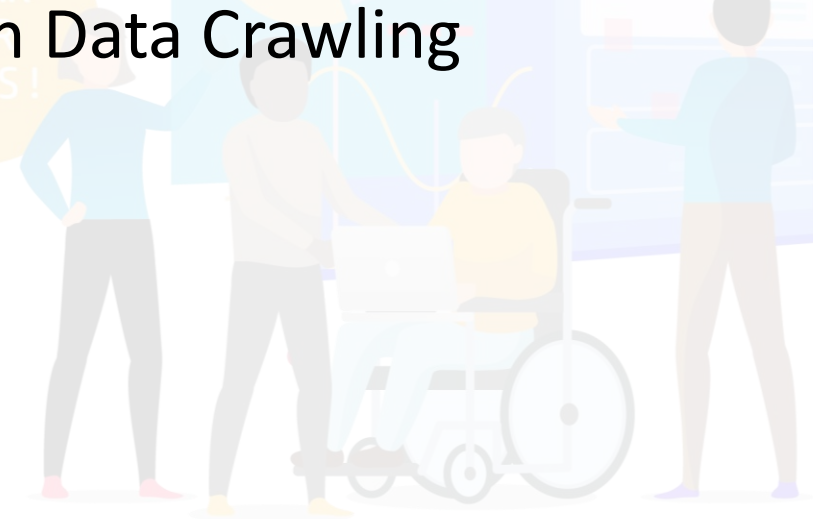


Daftar Isi

- Pengenalan Data Crawling
- Konsep dasar Data Crawling
- Web Crawling, sebagai contoh Data Crawling
 - Cara Kerja Web Crawler
 - Strategi Web Crawler
 - Arsitektur Web Crawler
 - Kebijakan Web Crawler
- Data Crawling dari Twitter
 - Konfigurasi Twitter untuk Data Crawling

TERBUKA
UNTUK
DISABILITAS

BREAK
LIMITS!



Pengenalan *Data Crawling* (1)

- Big Data adalah terminologi pada komputer yang merujuk kepada pengolahan data yang memiliki ukuran yang besar.
- Salah satu sumber yang dapat menyebabkan ukuran data yang besar adalah *data crawling*.
- *Data crawling* adalah teknik untuk mencari informasi dari berbagai sumber data. *Data crawling* akan melacak informasi ke setiap level sumber data yang dapat diakses.

Pengenalan *Data Crawling* (2)

- Sumber data crawling yang paling umum adalah “website”.
 - *Data crawling* yang bersumber dari website biasa disebut *web crawling*.
- *Data crawling* sangat berguna bagi perusahaan dalam mencari berbagai informasi yang tersebar di berbagai website.
- Terutama untuk perusahaan *search engine* yang menggunakan *web crawling* untuk membuat index web dari sebuah mesin pencarian di internet.

Konsep Dasar *Data Crawling* (1)

- *Data crawling* dilakukan menggunakan sebuah aplikasi *crawler* dengan konfigurasi tertentu.
- Aplikasi *crawler* adalah sebuah program yang dibuat untuk menjangkau semua sumber informasi dan melakukan aksi yang sudah ditentukan seperti memeriksa “kesegaran” dari sebuah informasi atau mengambil data dari sebuah sumber informasi.

Konsep Dasar *Data Crawling* (2)

- Langkah-langkah dasar dari sebuah proses *data crawling* adalah seperti terlihat pada gambar berikut:



Konsep Dasar *Data Crawling* (3)

- Langkah-langkah *Data Crawling*:
 - Menentukan sumber-sumber informasi
 - Langkah pertama adalah menentukan atau membuat daftar sumber-sumber informasi. Misalnya untuk *web crawling*, membuat daftar-daftar URL website yang akan diambil informasinya. Daftar website harus kredibel dan hindari website yang tidak mengizinkan *automated crawling* pada konfigurasi robots.txt atau di halaman TOS.
 - Mengkonfigurasi aplikasi *crawler*
 - Langkah kedua membutuhkan kemampuan teknis khususnya kemampuan pemrograman untuk memahami struktur data pada sumber informasi dan mengidentifikasi poin-poin yang bisa diambil dari sumber informasi tersebut sesuai dengan tugas yang sudah ditentukan.

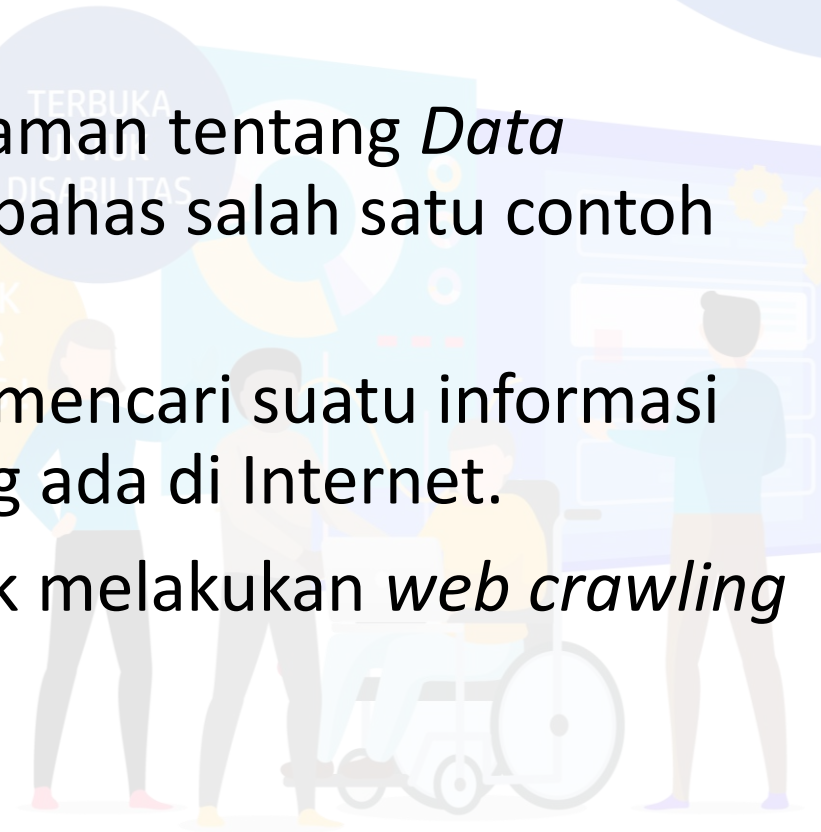
TERBUKA
UNTUK
DISABILITAS

Konsep Dasar *Data Crawling* (4)

- Melakukan *cleansing* dan menghilangkan duplikasi data
 - Data awal hasil *crawler* umumnya penuh dengan data-data anomali dan mengandung duplikasi informasi. Kondisi ini dapat mempengaruhi akurasi dari proses dan analisa data. Karena itu, langkah ini menjadi penting untuk membersihkan data dari data-data anomali serta data yang terduplikasi.
- Restrukturisasi data
 - Data yang didapat dari hasil *cleansing* dan penghilangan duplikasi, perlu diubah struktur-nya ke dalam skema yang dipahami oleh komputer. Dengan data yang terstruktur, akan mempermudah pemrosesan dan analisa lebih lanjut.

Web Crawlers (1)

- Untuk memperdalam pemahaman tentang *Data Crawling*, selanjutnya akan dibahas salah satu contoh yaitu *Web crawling*.
- *Web crawling* secara spesifik mencari suatu informasi dari sekumpulan website yang ada di Internet.
- Aplikasi yang digunakan untuk melakukan *web crawling* adalah *web crawlers*.

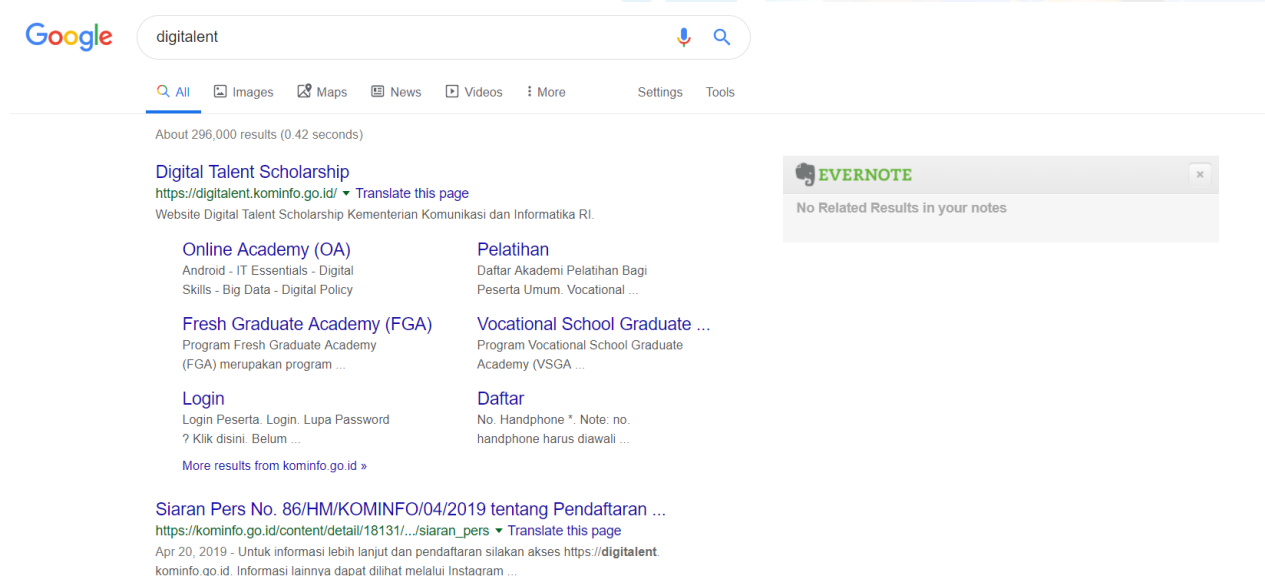


Web Crawlers (2)

- *Web crawler* adalah
 - sebuah proses atau program yang digunakan oleh mesin pencari untuk mengunduh halaman-halaman dari web untuk diproses lebih lanjut dalam membuat index web.
 - Index web kemudian digunakan untuk mempercepat proses pencarian informasi di Internet.
 - sebuah program atau script otomatis yang merambah website di internet dengan cara yang terstruktur.
 - juga dikenal sebagai aplikasi *web spider* dan *web robots*.
 - Istilah lain yang jarang digunakan: *ants*, *bots* dan *worms*.

Web Crawlers (3)

- *Web crawlers* sangat berguna karena:
 - Internet memiliki segudang website yang mengandung berbagai jenis informasi.
 - Mencari informasi yang relevan dari sekumpulan website di Internet membutuhkan mekanisme yang efisien.



Cara Kerja *Web Crawler* (1)

- *Web crawler* mengawali proses dari daftar URL website yang harus dikunjungi yang biasa disebut *seeds*.
 - Saat *web crawler* mengunjungi setiap website di dalam daftar tersebut, ia akan mengidentifikasi sebuah *hyperlinks* yang ada di dalam website tersebut dan menambahkannya ke dalam daftar URL yang telah dikunjungi, yang disebut *crawl frontier*.
- URL dari daftar *crawl frontier* kemudian menjadi *seeds* untuk iterasi kunjungan berikutnya berdasarkan sekumpulan konfigurasi yang sudah ditetapkan.

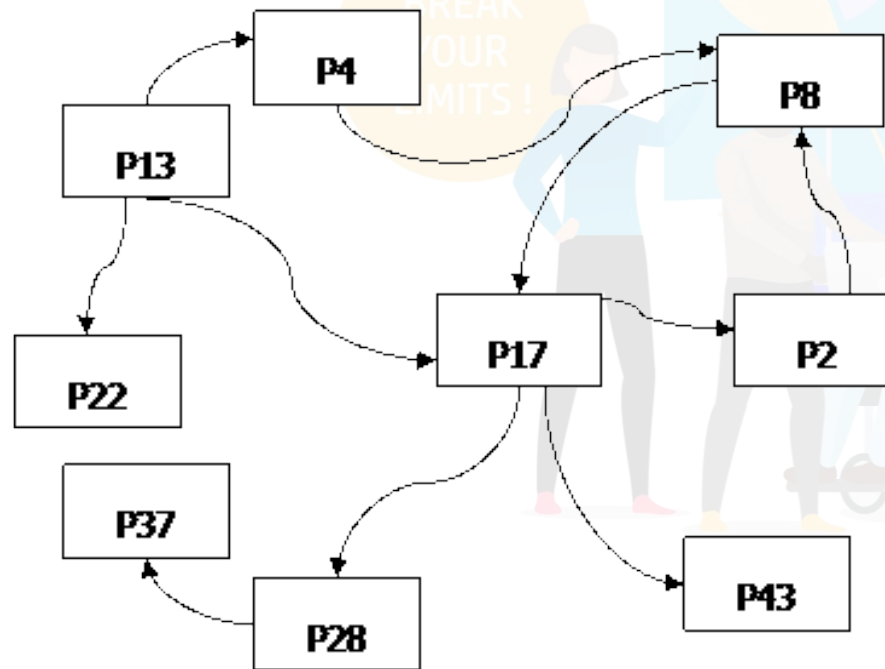
Cara Kerja *Web Crawling* (2)

- *Web crawlers* bekerja dengan algoritma sebagai berikut:

```
Initialize queue (Q) with initial set of known URL's.  
Until Q empty or page or time limit exhausted:  
    Pop URL, L, from front of Q.  
    If L is not an HTML page (.gif, .jpeg, .ps, .pdf,  
    .ppt...)  
        exit loop.  
    If already visited L, continue loop(get next url).  
        Download page, P, for L.  
    If cannot download P (e.g. 404 error, robot excluded)  
        exit loop,  
    else.  
        Index P (e.g. add to inverted index or store  
        cached copy).  
        Parse P to obtain list of new links N.  
        Append N to the end of Q.
```

Cara Kerja *Web Crawling* (3)

- Hasil *web crawler* biasanya berupa graph index seperti pada gambar berikut:

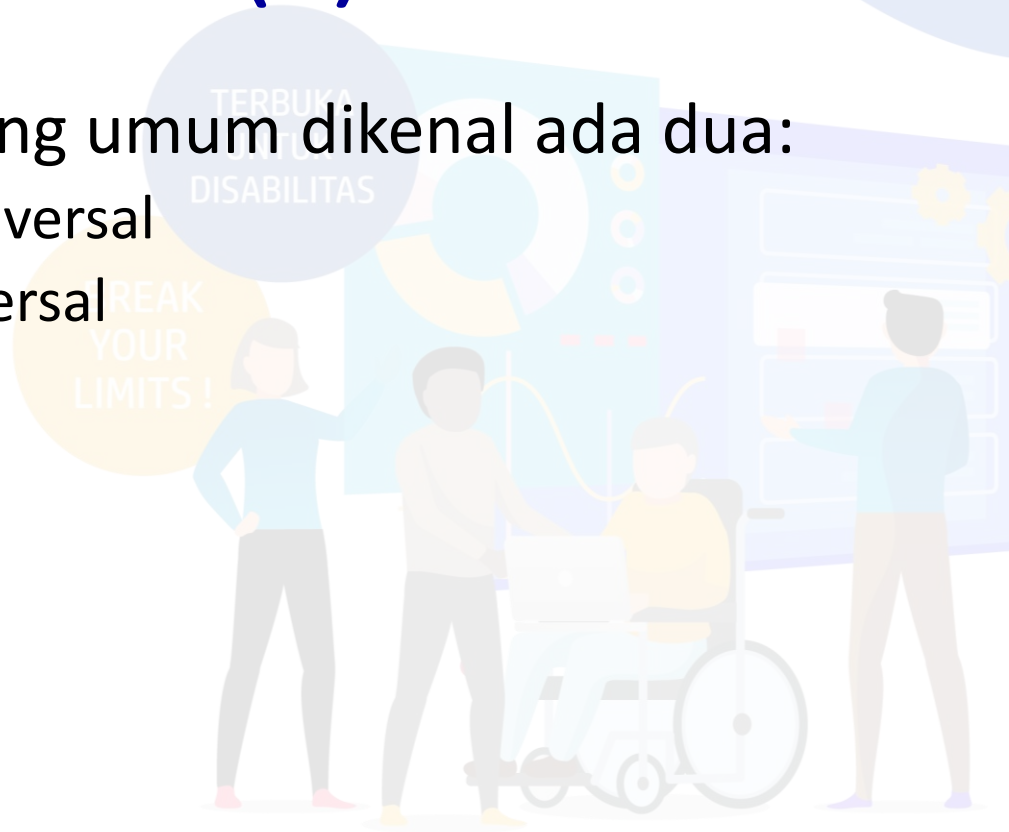


Strategi *Web Crawler* (1)

- Strategi *web crawler* juga bisa disebut sebagai algoritma aplikasi *web crawler*.
- Strategi *web crawler* memiliki asumsi:
 - Web adalah sebuah *directed graph* yang sangat besar, dengan dokumen sebagai *vertices* dan hyperlinks sebagai *edges*.
 - Analisa *directed graph* lebih lanjut menggunakan algoritma *graph traversal*.
 - Dalam algoritma *graph traversal*, *nodes* atau *vertices* digambarkan sebagai kotak dan *directed edges* digambarkan sebagai anak panah.

Strategi *Web Crawler* (2)

- Strategi *web crawler* yang umum dikenal ada dua:
 - Breadth-First Search Traversal
 - Depth-First Search Traversal



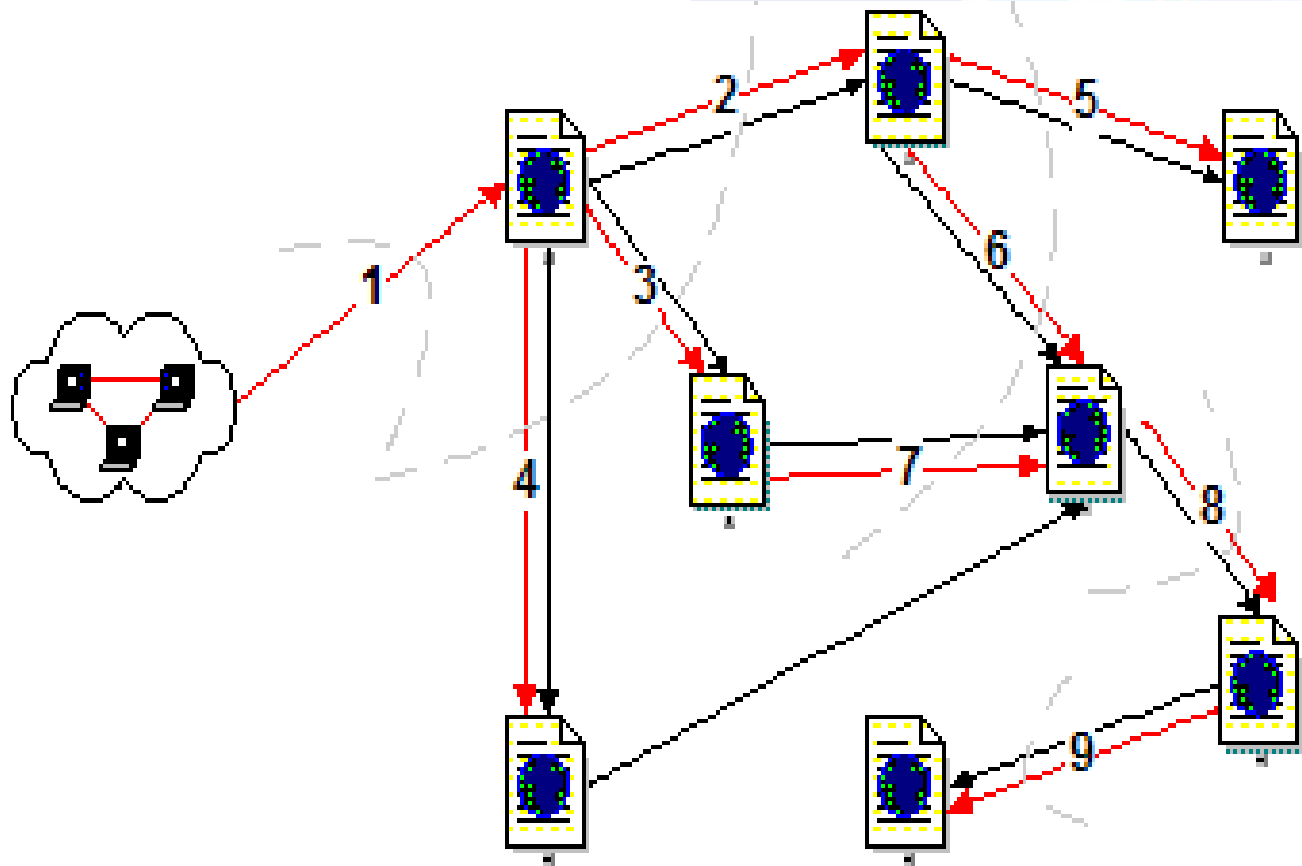
Strategi Web Crawler (3)

- Breadth-First Traversal
 - Dengan sembarang *graph* dan sekumpulan *seeds*, *graph* akan bisa dilalui dengan algoritma:
 1. Masukkan daftar URL di dalam *seeds* ke dalam antrian;
 2. Persiapkan daftar *nodes* yang telah dikunjungi (daftar ini awalnya adalah kosong);
 3. Selama antrian masih berisi data URL:
 - a. Hilangkan *node* pertama dari antrian;
 - b. Tambahkan *node* tersebut ke dalam daftar *node* yang telah dikunjungi.
 - c. Untuk setiap *edge* yang dimulai dari *node* tersebut:
 - i. Jika *node* pada akhir *edge* telah terdapat di dalam daftar *node* yang telah dikunjungi atau sudah ada di dalam antrian, jangan lakukan apa-apa terhadap *edge* tersebut;
 - ii. Sebaliknya, tambahkan *node* pada akhir *edge* ke bagian akhir dari antrian.

TERBUKA
UNTUK
SEMUAYOUR
LIMITS!

Strategi *Web Crawler* (4)

- Ilustrasi strategi breadth-first crawlers:

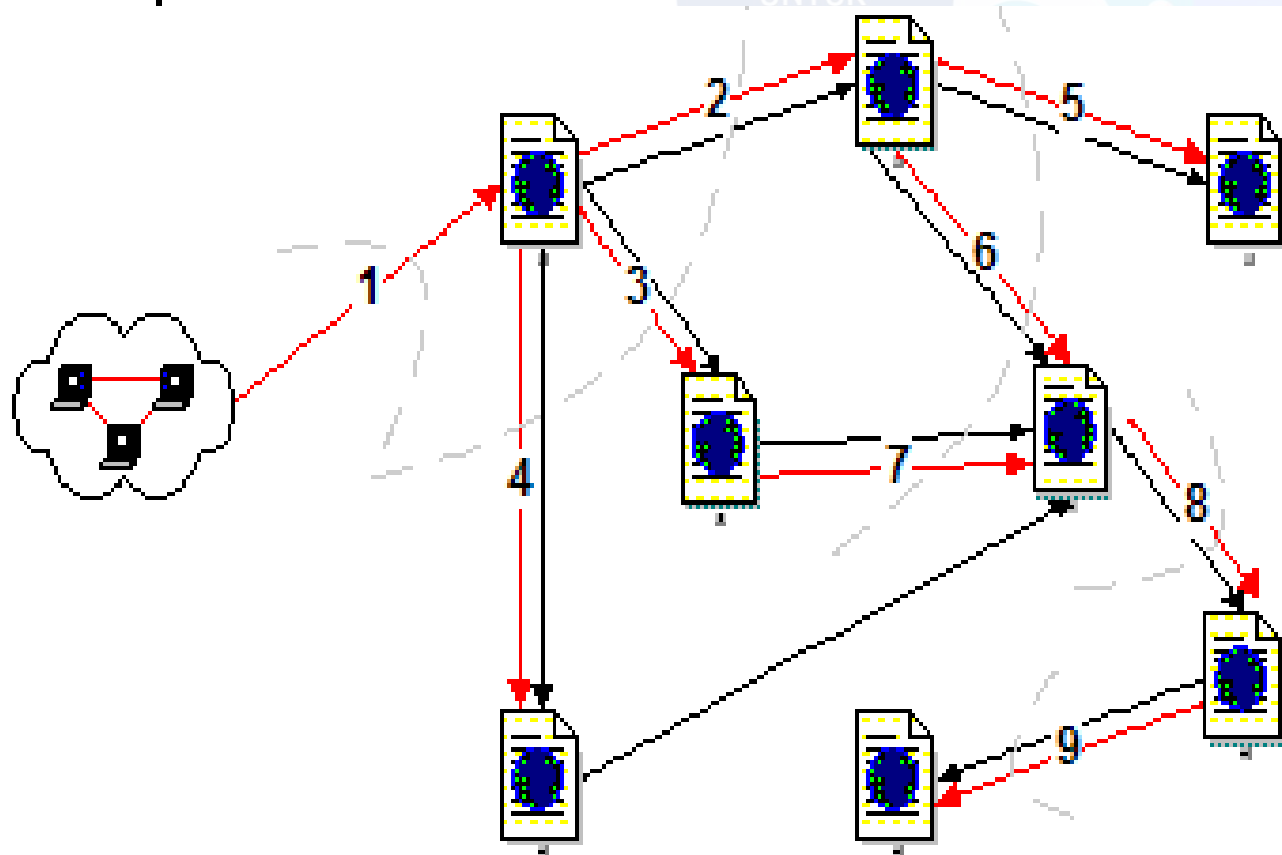


Strategi *Web Crawler* (5)

- Algoritma depth-first search (DFS):
 - Ambil link pertama yang belum dikunjungi dari halaman awal.
 - Kunjungi link tersebut dan ambil link pertama yang belum dikunjungi.
 - Ulangi langkah-langkah di atas hingga tidak ada lagi link yang belum dikunjungi.
 - Kunjungi link yang belum dikunjungi berikutnya yang terdapat di dalam level sebelumnya dan ulangi langkah kedua.

Strategi *Web Crawler* (6)

- Ilustrasi depth-first traversal:

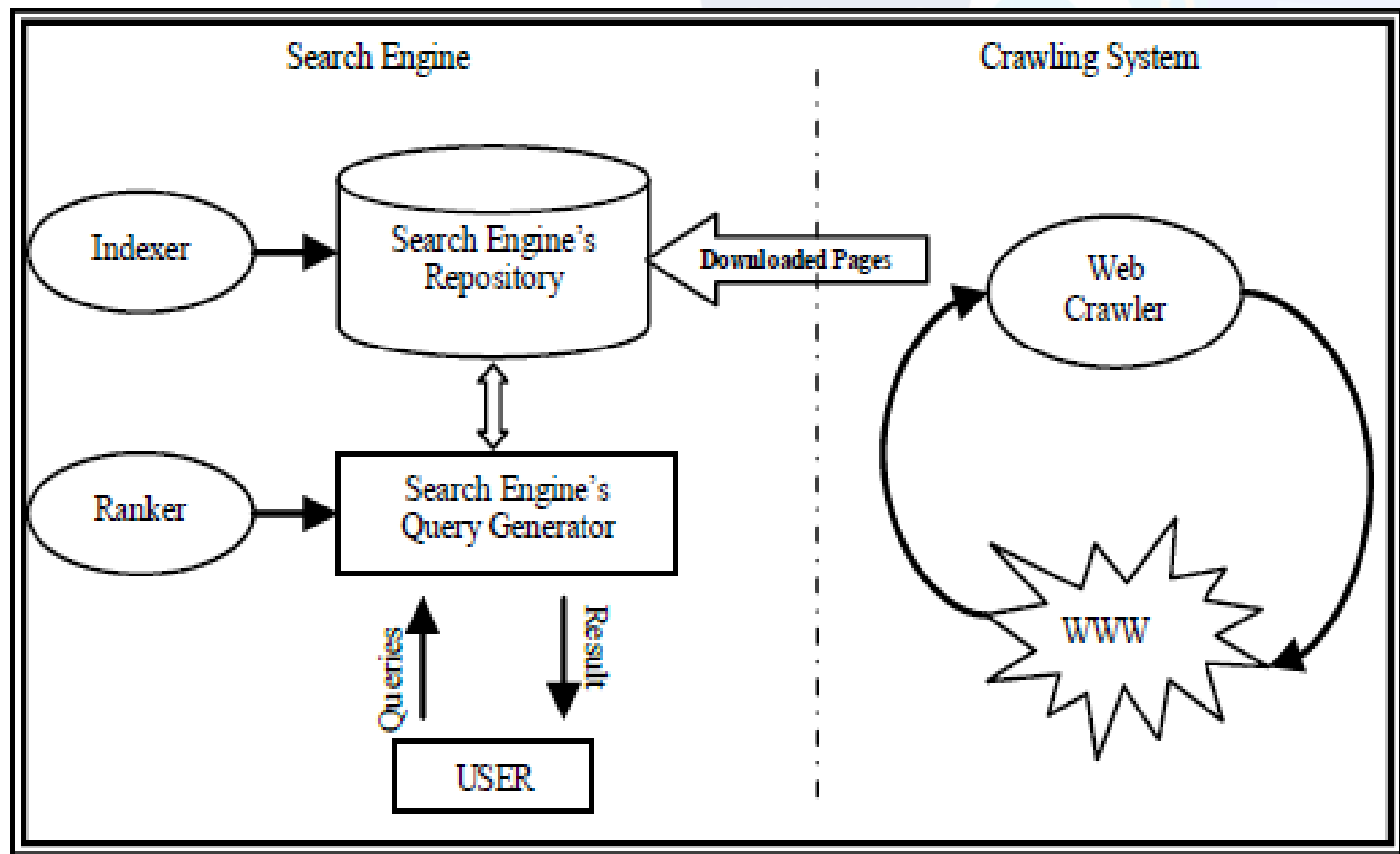


Strategi *Web Crawler* (7)

- Depth-First vs Breadth-First:
 - Depth-First bekerja pada satu *vertices* hingga mencapai sebuah *edge*
 - Tidak cukup bagus jika *edge* berada di *vertices* yang berbeda
 - Tidak lengkap atau optimal
 - Menggunakan ruang yang lebih sedikit di banding Breadth-First
 - Lebih sedikit *node* yang telah dikunjungi yang perlu di awasi
 - Ukuran lebih kecil
 - Breadth-First memeriksa semua alternative dengan lebih teliti
 - Lengkap dan optimal
 - Membutuhkan memory yang cukup besar

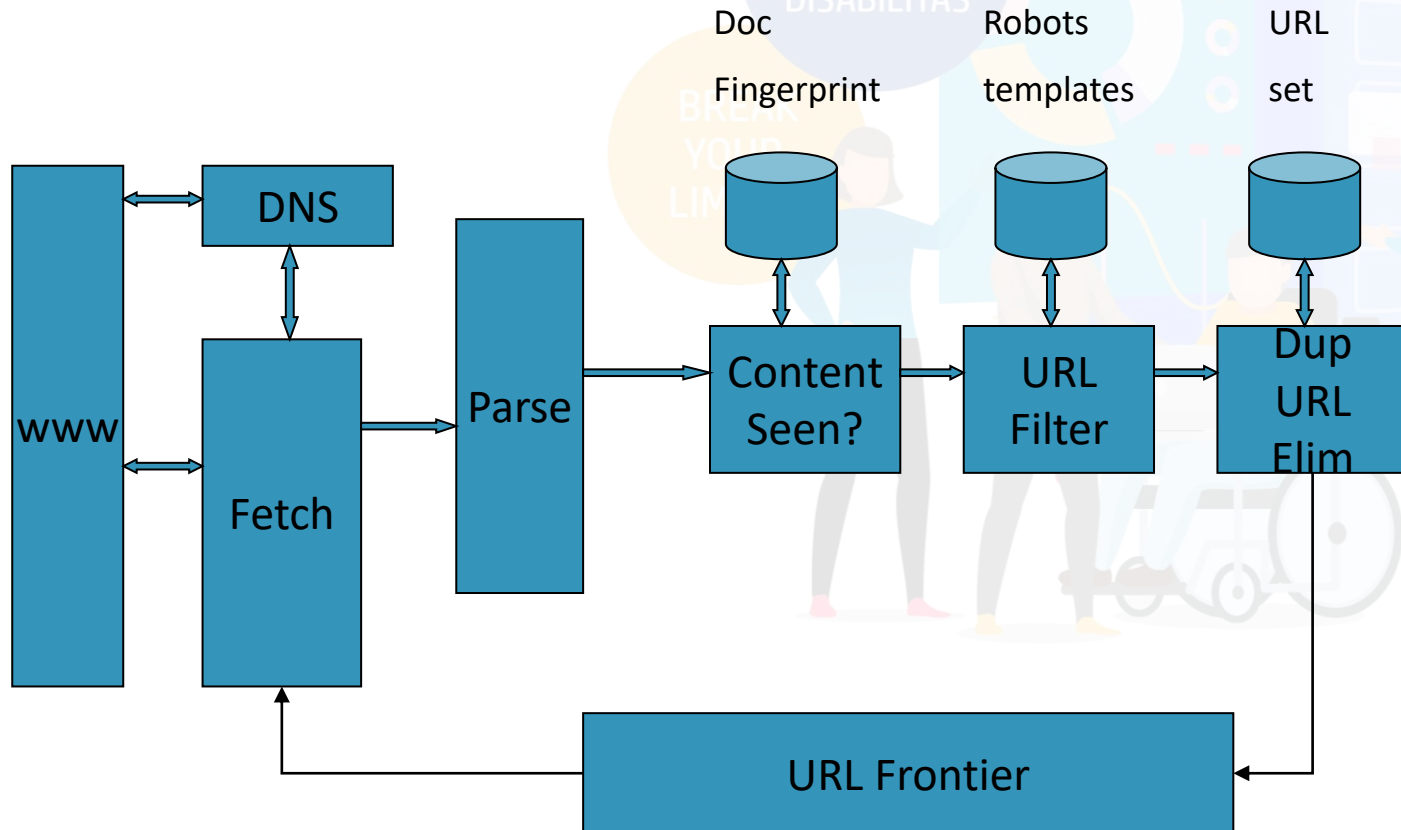
Arsitektur *Web Crawler* (1)

- Ilustrasi arsitektur *search engine*:



Arsitektur *Web Crawler* (2)

- Ilustrasi arsitektur *web crawler*:



Arsitektur *Web Crawler* (3)

- **URL Frontier:** mengandung URL yang belum dikunjungi. Awalnya, sebuah *seeds* disimpan pada URL Frontier dan sebuah *web crawler* memulai proses dari URL yang berada di dalam URL frontier.
- **DNS:** singkatan dari Domain Name Service yang memetakan sebuah domain name dengan alamat IP.
- **Fetch:** umumnya menggunakan protokol HTTP untuk mengakses URL.
- **Parse:** halaman web di urai dan mengambil informasi text dan link.
- **Content Seen?:** menguji apakah sebuah halaman web dengan isi yang sama telah diakses melalui URL yang lain.

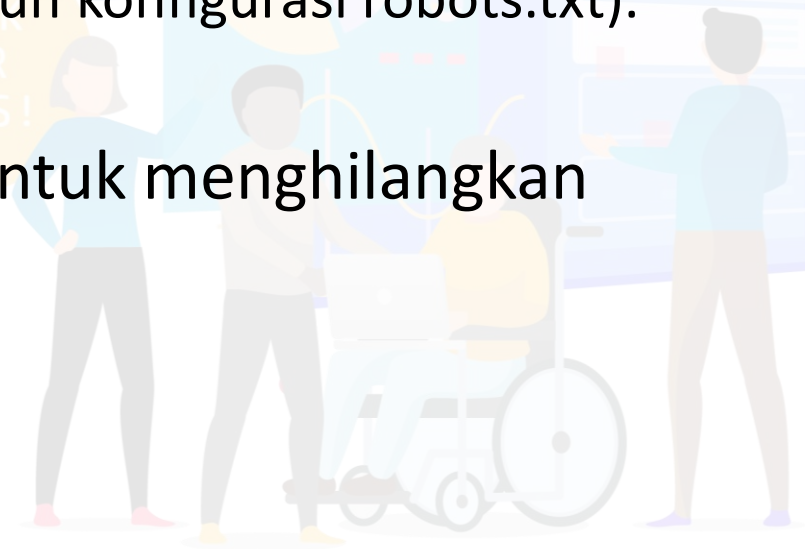
Arsitektur *Web Crawler* (4)

- **URL Filter:**

- Untuk memeriksa apakah URL yang terbaca harus dikeluarkan dari daftar URL Frontier (pengaruh konfigurasi robots.txt).
- URL harus dinormalisasi.

- **Dup URL Elim:** URL diperiksa untuk menghilangkan duplikasi.

TERBUKA
UNTUK
DISABILITAS
DON'T
BREAK
YOUR
LIMITS!

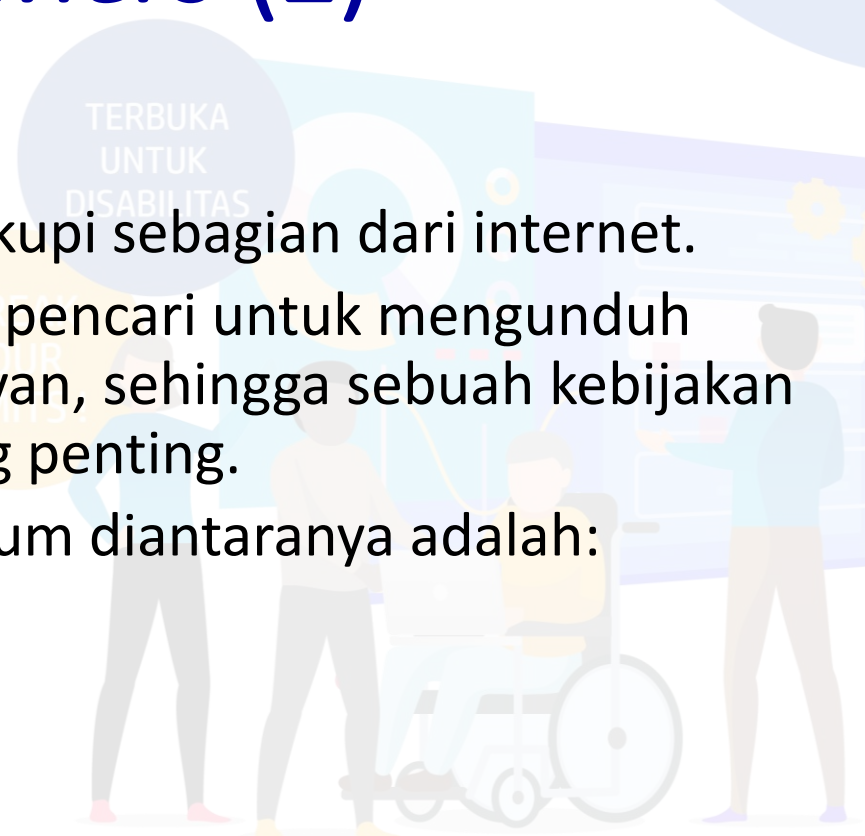


Kebijakan *Web Crawler* (1)

- Kebijakan *web crawler* terdiri dari:
 - Kebijakan *selection* yang menentukan halaman mana yang bisa diakses.
 - Kebijakan *re-visit* yang menentukan kapan sebuah halaman dikunjungi kembali untuk mendeteksi adanya pembaharuan isi.
 - Kebijakan *politeness* untuk menentukan bagaimana menghindari proses yang memberi kelebihan beban bagi website.
 - Kebijakan *parallelization* mengatur cara mengkoordinasi *web crawler* yang terdistribusi.

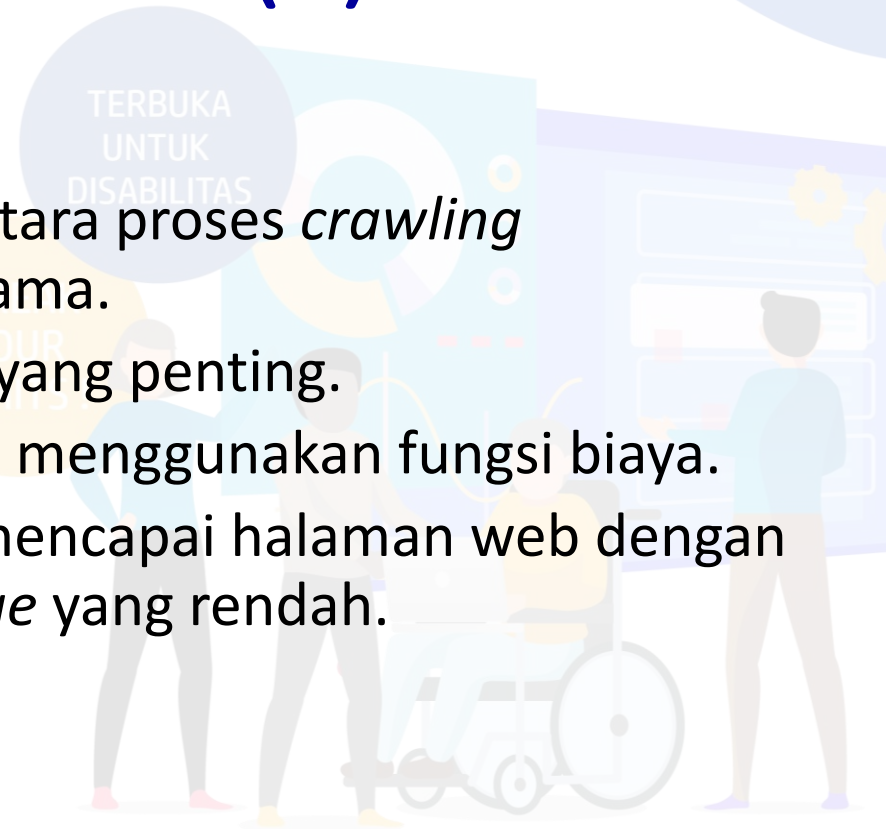
Kebijakan *Web Crawlers* (2)

- Kebijakan *Selection*:
 - Mesin pencari hanya melingkupi sebagian dari internet.
 - Hal ini membutuhkan mesin pencari untuk mengunduh halaman-halaman yang relevan, sehingga sebuah kebijakan *selection* memiliki peran yang penting.
 - Kebijakan *selection* yang umum diantaranya adalah:
 - *Restricting followed links*
 - *Path-ascending crawling*
 - *Focused crawling*
 - *Crawling the Deep Web*



Kebijakan *Web Crawlers* (3)

- Kebijakan *Re-visit*:
 - Web bersifat dinamis sementara proses *crawling* membutuhkan waktu yang lama.
 - Faktor biaya memiliki peran yang penting.
 - *Freshness* dan *Age*- biasanya menggunakan fungsi biaya.
 - Tujuan dari *crawler* adalah mencapai halaman web dengan *Freshness* yang tinggi dan *Age* yang rendah.
 - Dua kebijakan *re-visit*:
 - *Uniform Policy*
 - *Proportional Policy*



Kebijakan *Web Crawlers* (4)

- Kebijakan *Politeness*:
 - *Crawlers* bisa memiliki efek yang dapat melumpuhkan kinerja sebuah website.
 - Penggunaan *web crawlers* memiliki efek pada:
 - Sumber daya jaringan.
 - Kelebihan beban pada Server.
 - Kerusakan pada Server/Router.
 - Gangguan jaringan dan server.
 - Solusi terpisah dari masalah-masalah ini adalah *robots exclusion protocol*.

TERBUKA
UNTUK
DISABILITAS

BREAK
YOUR
LIMITS



Kebijakan *Web Crawlers* (5)

- Untuk mendukung kebijakan *Politeness*, penerapan teknisnya adalah dengan menggunakan *robot exclusion*.
 - Website dan halaman bisa menspesifikasi bahwa robot tidak boleh mengunjungi atau melakukan index pada area tertentu.
 - Dua komponen dari *robot exclusion*:
 - ***Robot Exclusion Protocol*** (robots.txt): website memberi spesifikasi daftar direktori yang tidak boleh diakses.
 - ***Robots META Tag***: Setiap halaman/dokumen memberi *meta tag* yang melarang robot untuk melakukan index atau mendeteksi link yang ada pada halaman tersebut.

Kebijakan *Web Crawlers* (6)

- Untuk mendukung kebijakan *Politeness*, penerapan teknisnya adalah dengan menggunakan *robot exclusion*.
 - Website dan halaman bisa menspesifikasi bahwa robot tidak boleh mengunjungi atau melakukan index pada area tertentu.
 - Dua komponen dari *robot exclusion*:
 - **Robot Exclusion Protocol** (robots.txt): website memberi spesifikasi daftar direktori yang tidak boleh diakses.
 - **Robots META Tag**: Setiap halaman/dokumen memberi *meta tag* yang melarang robot untuk melakukan index atau mendeteksi link yang ada pada halaman tersebut.

Kebijakan *Web Crawlers* (7)

- Admin website menaruh sebuah file “robots.txt” pada direktori utama (root) dari direktori web. Contohnya:
 - <http://www.ebay.com/robots.txt>
 - <http://www.Wikipedia.com/robots.txt>
- File “robots.txt” berisi daftar direktori yang tidak boleh diakses oleh robot tertentu (user-agent).
 - Contoh kode “robots.txt” yang melarang semua robot untuk keseluruhan website:

User-agent: *

Disallow: /

Kebijakan *Web Crawlers* (8)

- Kebijakan *Parallelization*
 - Aplikasi *crawler* menjalankan lebih dari satu proses secara parallel.
 - Tujuannya:
 - Memaksimalkan rata-rata waktu unduh
 - Mengurangi *overhead* dari parallelization
 - Untuk mencegah halaman yang sama diunduh berulang kali.
 - Sistem *crawling* membutuhkan kebijakan untuk menandai URL baru yang ditemukan selama proses *crawling* berlangsung.

TERBUKA
UNTUK
DISABILITAS

BREAK
YOUR
LIMITS!

Data Crawling dari Twitter (1)

- Contoh berikutnya mengenai *Data Crawling* yang sekarang paling banyak digunakan dalam algoritma *Machine Learning* adalah *Data Crawling* dari Twitter.
- Twitter adalah aplikasi jaringan social yang populer dimana penggunanya dapat membagi pesan-pesan pendek yang disebut *tweets*.
- Penggunaan Twitter sangat beragam dari sekedar saling berbagi ide, link, gambar dan video, komentar jurnalis pada event *live* hingga promosi produk.
- Twitter dapat mencapai 500 juta tweets dalam satu hari, yang merupakan data yang sangat besar untuk diolah.

Data Crawling dari Twitter (2)

- Cara mengolah data dari twitter tidak berbeda dari tahapan pada *data crawler*.
- Tahap pertama, seorang data analis harus memiliki beberapa *key* yang memungkinkan koneksi ke API yang disediakan oleh Twitter.
- Data analis menggunakan *key* tersebut pada bahasa pemrograman yang dipilih untuk menganalisa data-data twitter.

TERBUKA
IN
DISABILITAS

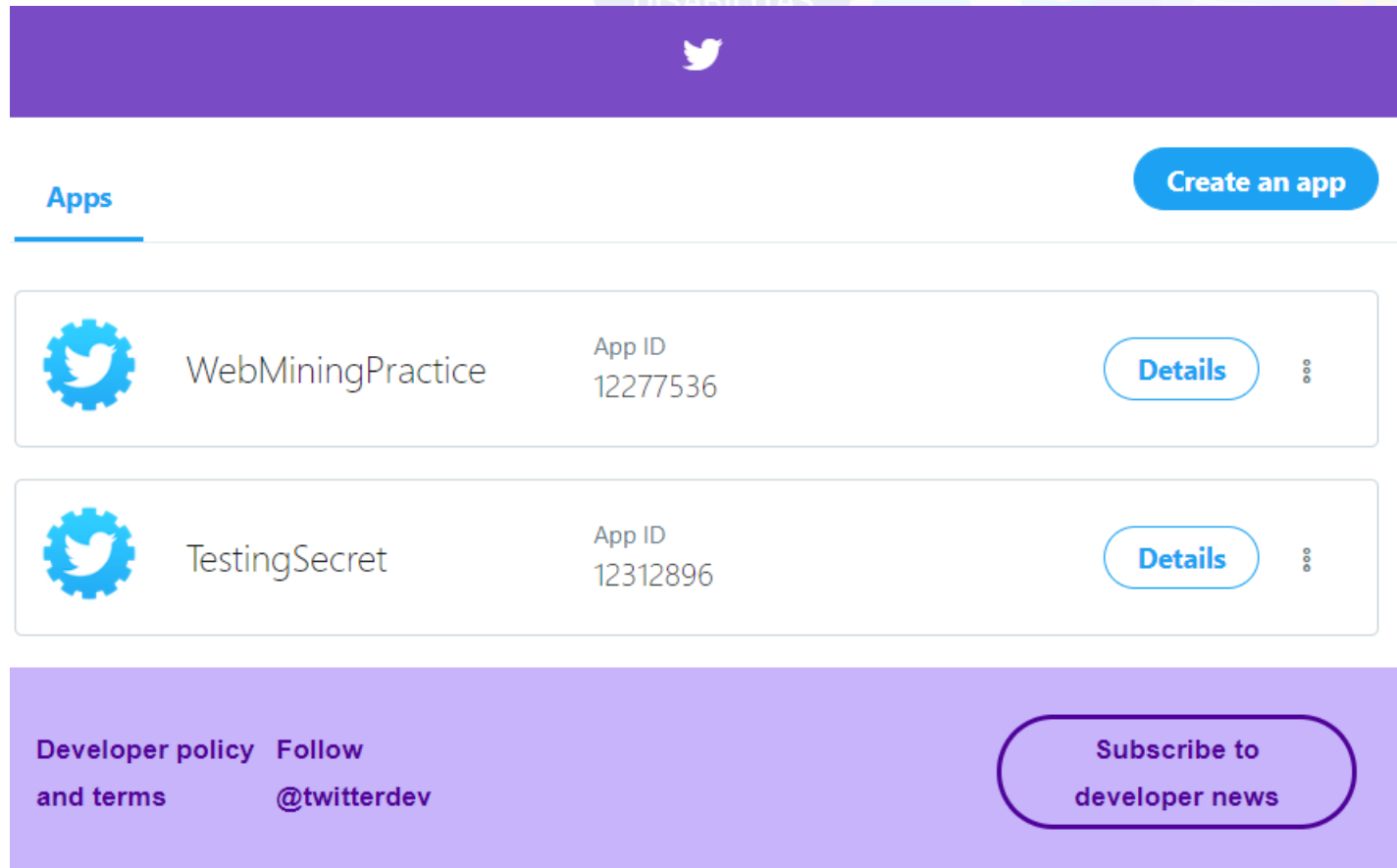
BREAK
THROUGH

Data Crawling dari Twitter (3)





- Untuk mendapatkan *access key* ke twitter, data analis harus membuat aplikasi yang berinteraksi dengan Twitter API.
- Langkah pertama adalah mendaftarkan aplikasi ke Twitter. Linknya adalah:
<https://developer.twitter.com/en/apps>
- Login menggunakan akun twitter atau buat akun twitter jika belum memilikinya.

Data Crawling dari Twitter (4)

- Tampilan dari halaman developer twitter adalah seperti berikut.



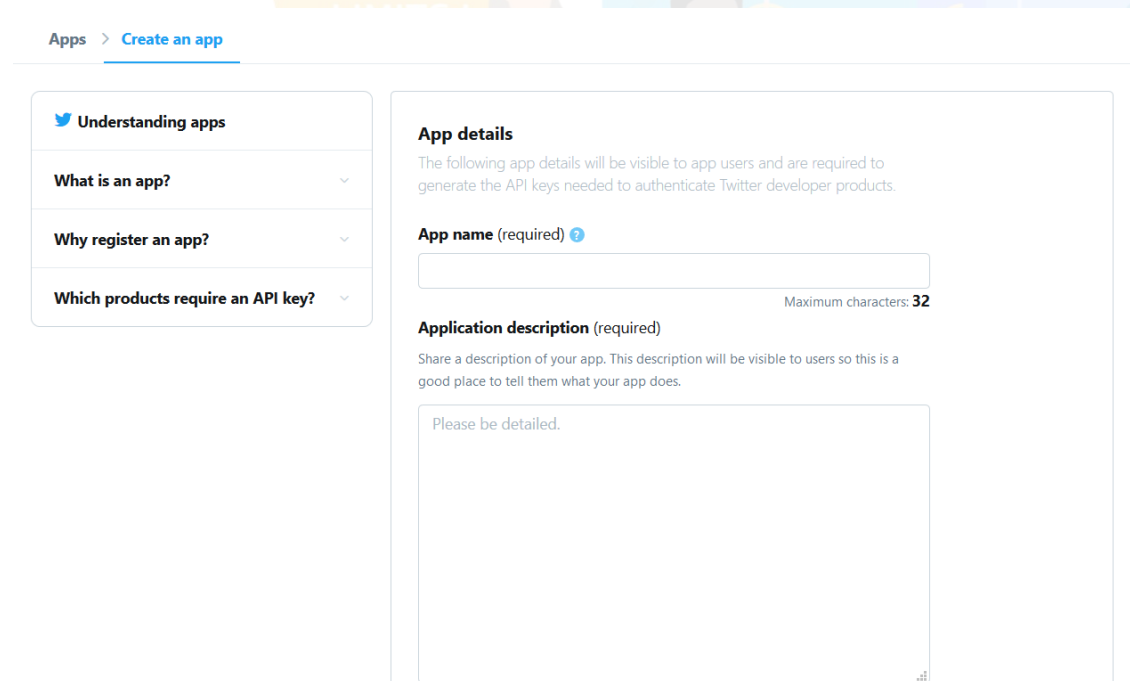
Apps [Create an app](#)

	WebMiningPractice	App ID 12277536	Details 
	TestingSecret	App ID 12312896	Details 

[Developer policy and terms](#) [Follow @twitterdev](#) [Subscribe to developer news](#)

Data Crawling dari Twitter (5)

- Pada halaman tersebut, klik link Create an app untuk mendaftar aplikasi yang akan mengakses data twitter.
- Halaman yang akan muncul adalah seperti berikut:



The screenshot shows the 'Create an app' page on the Twitter developer portal. The page is divided into two main sections: 'Understanding apps' on the left and 'App details' on the right.

Understanding apps

- What is an app?**
- Why register an app?**
- Which products require an API key?**

App details

The following app details will be visible to app users and are required to generate the API keys needed to authenticate Twitter developer products.

App name (required)

[Text input field]

Maximum characters: 32

Application description (required)

Share a description of your app. This description will be visible to users so this is a good place to tell them what your app does.

[Text area with placeholder text: Please be detailed.]

Data Crawling dari Twitter (6)


- Bila berhasil mendaftar aplikasi baru ke twitter, akan muncul halaman berikut.

Apps > [WebMiningPractice](#)

[App details](#) [Keys and tokens](#) [Permissions](#)

App details

Details and URLs

 **App icon**
App icon is default, click edit to upload.

App Name
WebMiningPractice

Description
Practicing Web Mining Source Code

Website URL
<http://twitter.com>

Sign in with Twitter
Enabled

Callback URL

[Edit](#) ▼

Data Crawling dari Twitter (7)

- Klik tab Keys and Token untuk melihat access key yang dibutuhkan untuk memungkinkan aplikasi yang dibuat mengakses data Twitter seperti pada gambar berikut:

Apps > [WebMiningPractice](#)

App details

Keys and tokens

Permissions

Keys and tokens

Keys, secret keys and access tokens management.

Consumer API keys

[Redacted] (API key)

[Redacted] (API secret key)

Regenerate

Access token & access token secret

[Redacted] (Access token)

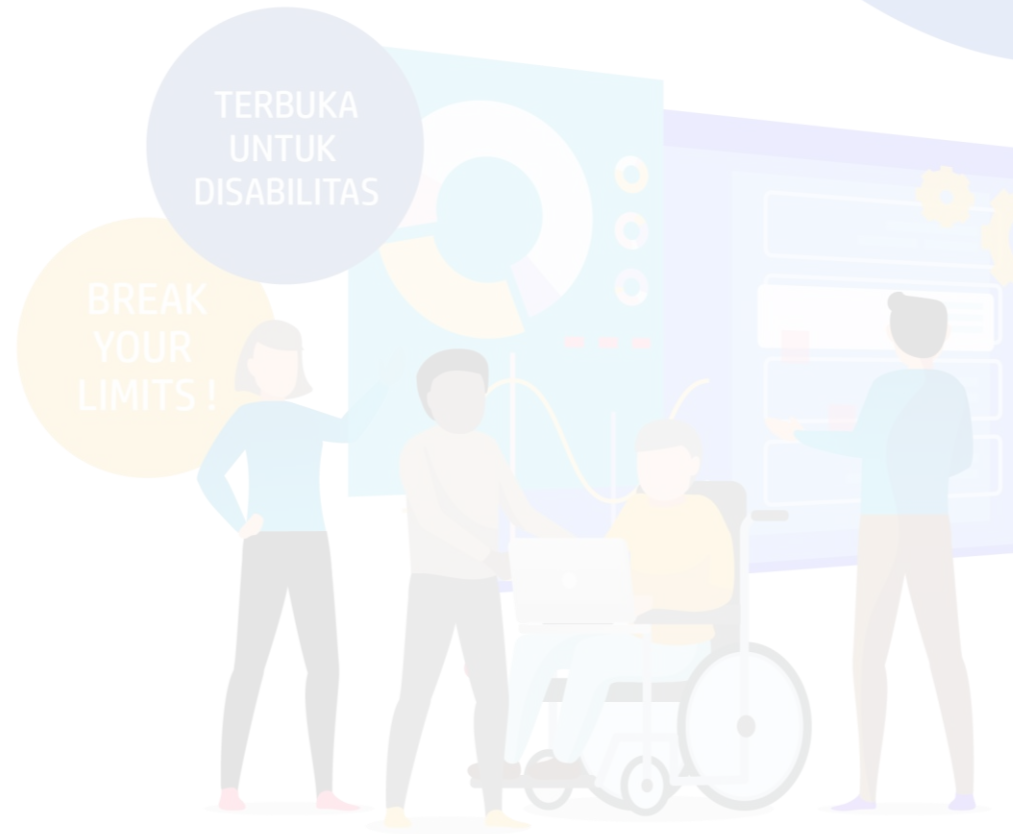
[Redacted] (Access token secret)

Read and write (Access level)

Revoke





Regenerate

- **Pertanyaan?**



IKUTI KAMI



-  [digitalent.kominfo](https://www.facebook.com/digitalent.kominfo)
-  [digitalent.kominfo](https://www.instagram.com/digitalent.kominfo)
-  [DTS_kominfo](https://twitter.com/DTS_kominfo)
-  [Digital Talent Scholarship 2019](https://www.telegram.com/join/digitalent.kominfo)

Pusat Pengembangan Profesi dan Sertifikasi
Badan Penelitian dan Pengembangan SDM
Kementerian Komunikasi dan Informatika
Jl. Medan Merdeka Barat No. 9
(Gd. Belakang Lt. 4 - 5)
Jakarta Pusat, 10110

