**Data Mining Review Hotel The Apurva Kempinski Bali Menggunakan BeautifulSoup**

**Dibuat Oleh: Stefannus Christian**

Link Github Project: <https://github.com/StefannusChristian/Data-Mining/tree/main/apurva-kempinski-bali-web-scraping>

Berikut merupakan link – link referensi belajar yang saya gunakan ketika mengerjakan projek ini:

1. Link Belajar Beautiful Soup: <https://www.youtube.com/watch?v=XVv6mJpFOb0>
2. Link Belajar WordCloud: <https://www.youtube.com/watch?v=HcKUU5nNmrs>

Disini saya menganalisis review penjunjung hotel yang termuat dalam website booking.com dengan link berikut ini: <https://www.booking.com/reviews/id/hotel/the-apurva-kempinski-bali.html?sid=d5b706fb1e392d7bdb21e2beb7949c62&aid=356980&order=featuredreviews&label=gog235jc-1FCA0oaEIZdGhlLWFwdXJ2YS1rZW1waW5za2ktYmFsaUgzWANoaIgBAZgBMbgBF8gBDNgBAegBAfgBAogCAagCA7gCnNKLogbAAgHSAiRhNTZlZDcxNS0yNTIzLTRmY2QtOTIwMy05MmRhYTVlYzdkOWLYAgXgAgE&hp_nav=0&page=1&customer_type=total&rows=75&old_page=0&r_lang=en>

Review – review tulisan di kotak merah pada gambar dibawah ini adalah review yang akan saya analisis.

Graphical user interface, application

Description automatically generated

Langkah – Langkah:

1. Install Library Python yang dibutuhkan

Jalankan ketiga perintah dibawah ini pada untuk menginstall library beautifulsoup yang akan digunakan untuk scraping, menginstall library wordcloud yang akan digunakan untuk membuat wordcloud dari hasil scraping, dan menginstall library lxml yang dibutuhkan untuk memproses html.

* *pip install beautifulsoup4*
* *pip install wordcloud*
* *pip install lxml*

1. Import Library yang diperlukan

Text

Description automatically generated

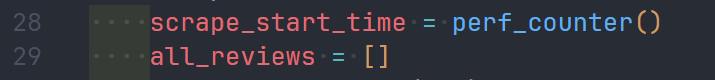
* Library requests method get digunakan untuk mengambil text dari situs booking.com.
* Library re (regex) method sub digunakan untuk membersihkan text hasil dari scraping seperti membuang spasi berlebih pada text hasil scraping.
* Library BeautifulSoup akan digunakan untuk web scraping.
* Library WordCloud akan digunakan untuk membuat wordcloud.
* Library time method perf\_counter akan digunakan untuk menghitung waktu eksekusi program seperti waktu eksekusi scraping per page dan juga waktu eksekusi membuat word cloud.

1. Scraping Web Booking.com

Text

Description automatically generated

Disini saya membuat function scrape\_reviews() untuk scraping web booking.com untuk hotel Apurva Kempinski Bali. Saya akan menjelaskan kode saya line per line dengan memberikan terlebih dahulu cuplikan kode kemudian menjelaskan apa yang kode tersebut lakukan.

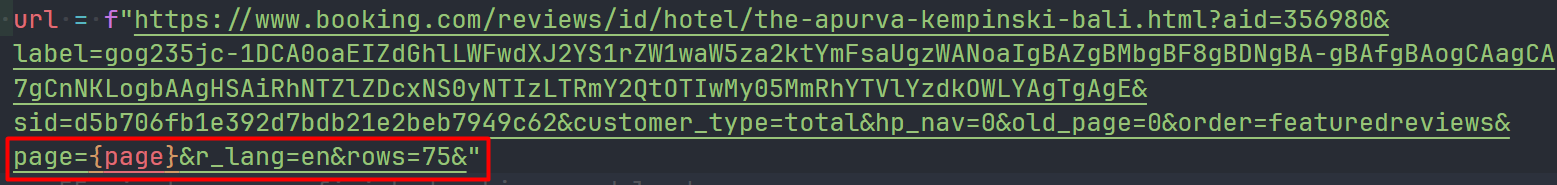


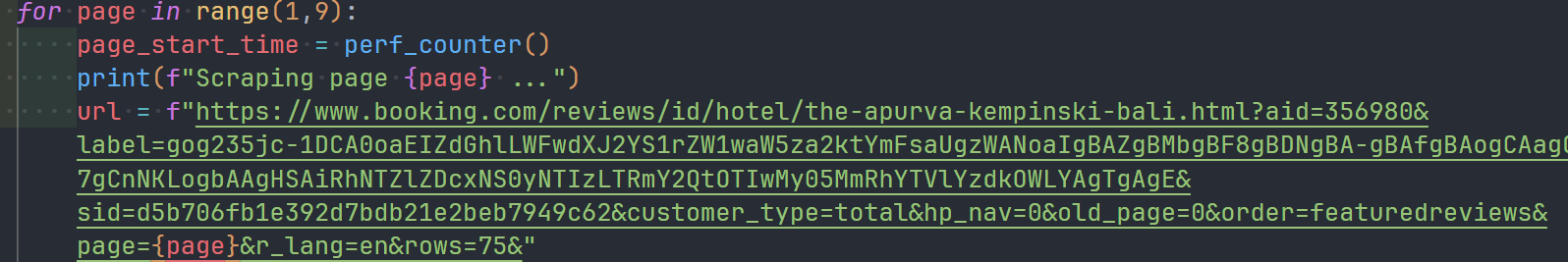
Variabel scrape\_start\_time akan mencatat waktu Ketika function scrape\_review dijalankan. Variabel ini nantinya akan digunakan untuk menghitung waktu yang digunakan untuk menjalankan scraping. Variabel all\_reviews digunakan untuk menyimpan seluruh review – review hasil scraping.

Graphical user interface, text, application

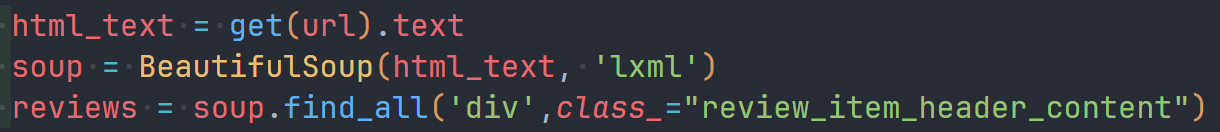
Description automatically generated

Terdapat 8 page review pada website ini. Website berikutnya dapat diakses dengan menekan next page seperti gambar diatas. Tetapi disini, saya menemukan bahwa URL yang digunakan untuk page berikutnya dapat diubah dari bagian url. Dapat dilihat dari gambar dibawah ini bahwa saya tinggal mengganti url web pada bagian {page} dan reviewnya pun akan berubah ke review halaman {page}.





Maka dari itu karena adanya 8 page pada website ini, saya melakukan for loop untuk mengiterasi dari page 1 – 8. Variabel page\_start\_time digunakan untuk menghitung start waktu dari ketika halaman {page} mulai di scrape dan nantinya akan digunakan untuk menghitung waktu yang dibutuhkan untuk scraping halaman {page}.



Disini variable html\_text digunakan untuk mendapatkan struktur html dari url yang diberikan. Variabel soup digunakan untuk menginisialisasi BeautifulSoup yang akan digunakan untuk scraping. Saya tidak akan menjelaskan secara detail apa yang dilakukan kedua variabel diatas, penjelasan lebih lanjut dapat ditonton [disini](https://www.youtube.com/watch?v=XVv6mJpFOb0).

Cara kerja beautifulsoup (bs) adalah bs dengan bantuan lxml (parser) akan mencari tag atau class dari elemen yang ingin di scrape. Disini, menggunakan dev tools browser untuk inspect element yang ingin dicari apa nama classnya dan class tersebut berada pada tag apa.

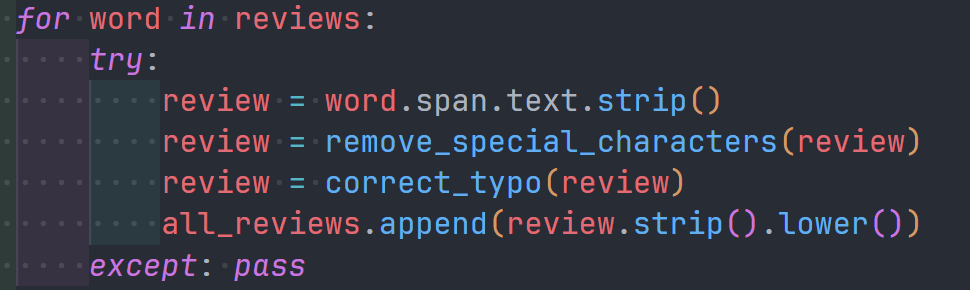
Graphical user interface

Description automatically generated with medium confidence

Dapat dilihat bahwa kata review yang ingin saya ambil yaitu Exceptional berada pada di tag span dengan parent tag div dengan class review\_item\_header\_content. Maka cara saya mendapatkan seluruh review adalah dengan mencari seluruh div tag dengan class review\_item\_header\_content. Berikut kode untuk melakukan hal ini.



Tetapi disini variabel reviews belum secara spesifik mendapatkan kata review yang diinginkan. Reviews disini bentuknya adalah list yang menyimpan seluruh div dengan class review\_item\_header\_content.



Berikut cara untuk mendapatkan kata review yang diinginkan. Iterasilah list reviews dan berdarkan struktur kode html yang telah saya berikan diatas, review word nya terdapat pada span sehingga kita perlu mengakses span tersebut dengan cara word.span. Kemudian .text digunakan untuk mendapatkan kata dari span tersebut dan .strip() digunakan untuk menghilangkan spasi berlebih dari string tersebut. Kemudian review tersebut akan dibersihkan dan di append ke list all\_reviews. Saya menggunakan function lower() untuk membuat review menjadi huruf kecil.

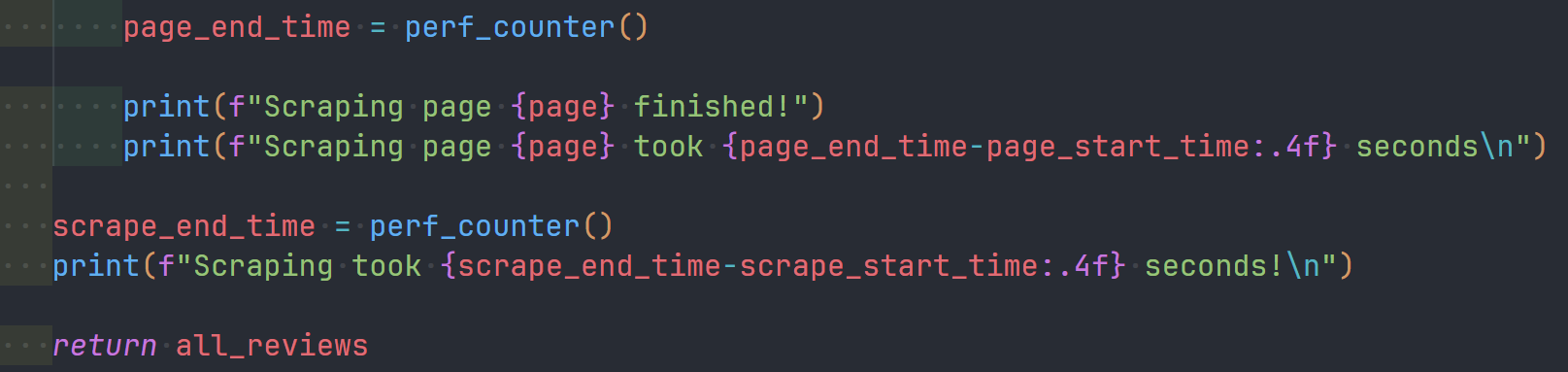
Text

Description automatically generated

Text

Description automatically generated

Kedua function diatas saya apply untuk membersihkan review sebelum dimasukkan kedalam list all\_reviews. Saya replace tanda tanda baca seperti titik,koma, dll. Saya juga membenarkan typo yang saya temukan di review tersebut seperti pleasant yang seharusnya pleasant.



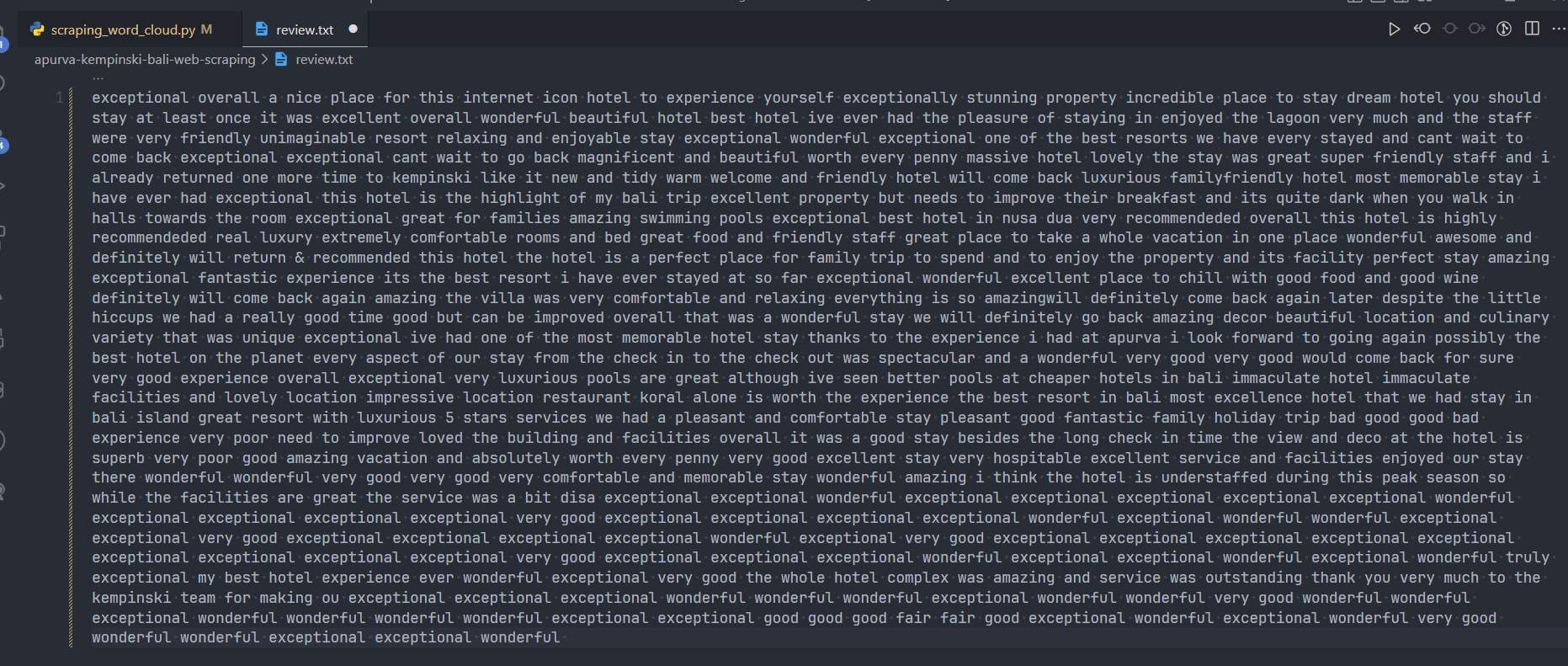
Kemudian variabel page\_end\_time digunakan sebagai tanda bahwa scraping page tersebut sudah selesai. Kemudian cara untuk menghitung berapa waktu yang digunakan untuk scraping satu page adalah dengan mengurangi waktu akhir dengan waktu awal. Begitu juga cara untuk menghitung waktu total yang digunakan untuk scraping seluruh page. Function ini kemudian akan return all\_reviews yang isinya adalah review words yang sudah di bersihkan.

1. Menyimpan hasil scraping ke file txt

Text

Description automatically generated

Untuk menconvert list all\_reviews menjadi word cloud, list all\_reviews perlu dibuat menjadi file txt terlebih dahulu. Yang saya lakukan pertama adalah membersihkan lagi list all\_reviews dengan menghilangkan NaN values, kemudian saya menggunakan method join untuk menggabungkan isi dari list tersebut dan membuatnya menjadi satu string Panjang (paragraph). Kemudian saya menggunakan method regex yaitu sub untuk menghilangkan spasi spasi dobel yang ada pada paragraph tersebut. Setelah itu saya convert paragraph tersebut ke file txt. Hasil dari file txt nya adalah sebagai berikut.



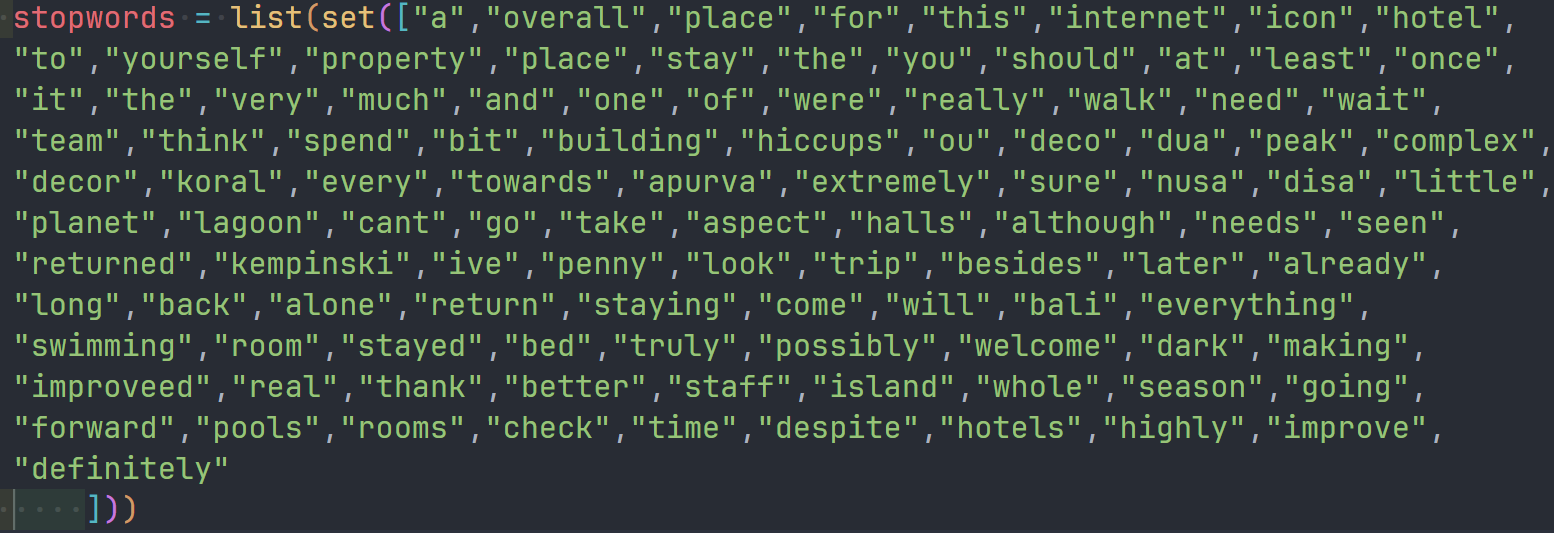
File txt inilah yang akan digunakan untuk membuat wordcloud.

1. Membuat wordcloud

Graphical user interface, text

Description automatically generated

Saya menggunakan library wordcloud pada python untuk membuat wordcloud. Untuk referensi nya saya belajar [disini](https://www.youtube.com/watch?v=HcKUU5nNmrs). Stopwords digunakan untuk memfilter kata – kata apa saja yang tidak diinginkan pada wordcloud. Ketika menginstall wordcloud, default STOPWORDS otomatis akan terinstall juga tetapi default STOPWORDS yang disediakan ini tidak cukup sehingga saya menambahkan stopwords berikut ini secara manual agar kata – kata ini tidak ditampilkan pada wordcloud.



Output program di console ketika di run adalah sebagai berikut.

Text

Description automatically generated

Hasil WordCloud

Text

Description automatically generated