**Pengenalan Dashboard**

Untuk melengkapi pengetahuan Anda, pada beberapa materi berikutnya, kita akan membahas berbagai hal tentang pengembangan dashboard. Di dunia industri, keahlian dalam membuat dashboard merupakan salah satu skill yang harus dimiliki oleh seorang praktisi data. Nah, bagi Anda yang belum pernah mendengar tentang dashboard, jangan khawatir karena pada materi ini kita akan membahasnya dari dasar.

**Apa itu Dashboard?**

Pernahkah Anda mengendarai mobil atau motor? Jika pernah, tentunya Anda sudah tidak asing lagi dengan gambar di bawah ini, bukan?



Bagi Anda yang belum tahu, gambar di atas merupakan tampilan dashboard mobil. Seperti yang Anda lihat, ia terdiri dari berbagai panel indikator serta speedometer, odometer, fuel meter, dll. Seluruh hal tersebut mampu memberikan kita gambaran terkait keadaan dari kendaraan tersebut. Nah, itulah contoh sederhana dari sebuah dashboard.

Sama halnya dengan contoh di atas, dashboard juga banyak digunakan sebagai media untuk memonitori berbagai metrik yang dianggap penting untuk perkembangan sebuah perusahaan. Selain itu, ia juga merupakan salah satu komponen penting dalam kultur *data-driven decision making*. Umumnya, sebuah dashboard terdiri dari beberapa visualisasi data yang interaktif dan saling berhubungan satu sama lain [13].

“*Hmm…*  apa yang dimaksud dengan visualisasi data yang interaktif?” Sederhananya, visualisasi data yang interaktif akan memungkinkan pengguna untuk mengontrol hal yang ingin ditampilkan dalam sebuah visualisasi data. Hal ini tentunya memberikan kebebasan terhadap user untuk fokus ke bagian yang menurut mereka penting.

Nah, pada beberapa materi ke depan kita akan belajar cara membuat sebuah dashboard. Namun, sebelum itu, mari kita lihat berbagai hal yang perlu dipertimbangkan sebelum membuat sebuah dashboard.

**Pertimbangan dalam Membuat Dashboard**

Saat ini, kita telah memahami apa itu dashboard. Namun, sebelum belajar cara membuat dashboard, kita perlu memahami berbagai hal penting dalam membuat dashboard yang efektif.

Dashboard yang efektif merupakan kunci utama dalam kultur data-driven decision making. Ia umumnya memiliki atau menampilkan metrik yang jelas serta mampu menyampaikan informasi yang relevan dengan kebutuhan audiens atau pengguna. Berikut merupakan beberapa hal yang harus diperhatikan dalam membuat dashboard yang efektif [14].

* **Memahami audiens**  
  Untuk membuat dashboard yang efektif, kita harus mempertimbangkan audiens dan kebutuhannya. Oleh karena itu, pada proses pembuatan dashboard kita harus memahami siapa target user atau audiens serta goals atau kebutuhan yang ingin mereka capai dari dashboard tersebut.
* **Pertimbangkan ukuran tampilan**  
  Sejalan dengan poin pertama, kita juga harus mempertimbangkan ukuran tampilan dashboard yang sesuai dengan *device* atau perangkat yang sering digunakan oleh target user atau audiens untuk melihat dashboard yang Anda buat.
* **Manfaatkan sweet spot**  
  Anda bisa menggunakan sweet spot untuk menampilkan informasi penting dari sebuah dashboard. Tentunya Anda perlu melakukan riset terlebih dahulu untuk mengetahui *sweet spot* yang tepat untuk kebutuhan Anda. Sebagai contoh, umumnya, bagian kiri atas merupakan *spot* pertama yang dilihat dalam sebuah halaman web. Nah, kita bisa menempatkan informasi terkait metrik penting seperti KPI (*key performance indicators*) pada spot tersebut.
* **Pertimbangkan load time**  
  Load time merupakan salah hal yang penting untuk kita pertimbangkan guna menjaga pengalaman user atau audiens. Oleh karena itu, kita harus memastikan dashboard yang dibuat memiliki load time yang relatif singkat.
* **Batasi komponen visual yang ditampilkan**  
  Sama halnya ketika kita membuat visualisasi data, pada proses pengembangan dashboard juga harus mempertimbangkan jumlah komponen visual yang akan ditampilkan. Komponen visual yang terlalu banyak tentunya akan mengganggu fokus dan perhatian user. Secara umum, kita disarankan untuk hanya menampilkan dua atau tiga visualisasi data. Selain itu, kita juga perlu menghindari penggunaan komponen visual yang tidak mendukung dalam penyampaian informasi penting.

Oke, itulah kelima poin yang harus kita pertimbangkan guna membuat dashboard yang efektif. Semoga paparan tersebut dapat membantu Anda ketika membuat sebuah dashboard nantinya.

Untuk membuat sebuah dashboard, tentunya kita membutuhkan sebuah tool, bukan? Nah, pada materi selanjutnya kita akan berkenalan dengan tool yang bisa digunakan untuk membuat sebuah dashboard.

**Pengenalan Streamlit**

Pada materi sebelumnya, kita telah membahas sedikit tentang Dashboard. Sekarang saatnya kita berkenalan dengan tool yang akan kita gunakan untuk membuat dashboard. Tentunya Anda sudah bisa menebak nama tool tersebut, bukan?



Streamlit merupakan salah satu*open-source web app framework* untuk bahasa pemrograman Python. Ia memungkinkan kita membuat web app yang baik dan interaktif dalam waktu yang singkat. Selain itu, ia juga kompatibel dengan berbagai library populer seperti NumPy, pandas, matplotlib, dll. Inilah yang menjadi alasan streamlit sering digunakan oleh oleh para praktisi data dan machine learning.

Apakah Anda sudah mulai tertarik untuk belajar streamlit? Yuk, kita mulai dengan menginstalnya!

**Menginstal Streamlit**

Seperti biasa, sebelum mulai menggunakan streamlit tentunya kita perlu menyiapkan *environment*-nya terlebih dahulu. Berikut merupakan tahapan untuk melakukannya.

1. Untuk menggunakan streamlit, kita perlu menulis kode Python dalam berkas**.py**. Oleh karena itu, Anda perlu menginstal code editor, seperti [Visual Studio Code](https://code.visualstudio.com/download), [Sublime Text](https://www.sublimetext.com/3), atau [PyCharm](https://www.jetbrains.com/pycharm/download/#section=windows) untuk mempermudah dalam menulis kode.
2. Selanjutnya, Anda perlu mengaktifkan virtual environment sebelumnya yang telah kita gunakan dalam materi latihan. Aktifkan virtual environment dengan menjalankan perintah berikut.

|  |
| --- |
| conda activate main-ds |

1. Tahap berikutnya adalah menginstal streamlit. Berikut merupakan perintah yang bisa Anda gunakan untuk menginstal streamlit.

|  |
| --- |
| pip install streamlit |

**Membuat "Hello, world!" App dengan Streamlit**

Selamat, pada tahap ini Anda telah berhasil menginstal streamlit. Untuk memastikan proses instalasinya berjalan dengan lancar, kita akan membuat sebuah “Hello, world!” app dengan streamlit.

Berikut merupakan tahapan dalam membuat “Hello, word!” app dengan streamlit.

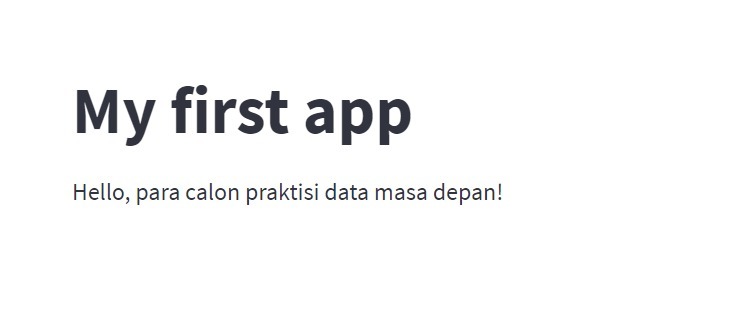
1. Buatlah sebuah folder baru bernama **hello-world**.
2. Bukalah folder tersebut dengan code editor favorit Anda.
3. Berikutnya, buatlah sebuah berkas python dengan nama **hello-world.py**.
4. Buka berkas python tersebut, lalu tambahkan kode berikut.

|  |
| --- |
| import streamlit as st    st.write(  """  # My first app  Hello, para calon praktisi data masa depan!  """  ) |

1. Untuk menjalankan kode tersebut, Anda perlu membuka terminal, command prompt atau powershell. Arahkan path-nya ke dalam folder **hello-world** yang sebelumnya telah dibuat. Lalu, jalankan perintah berikut.

|  |
| --- |
| streamlit run hello-world.py |

Jika seluruh proses di atas berjalan dengan lancar, Anda akan menemukan tampilan web app seperti berikut.



**Basic Element dalam Streamlit**

Sebelumnya, Anda telah berhasil menginstal dan membuat web app sederhana menggunakan streamlit. Untuk melengkapi pengetahuan Anda tentang streamlit, kita akan mengenal berbagai basic element yang terdapat dalam streamlit pada materi kali ini.

Sebagai sebuah web app framework yang andal, streamlit telah menyediakan banyak pilihan element, widget, layout serta container untuk memastikan kita dapat membuat web app atau dashboard yang menarik dan interaktif. Nah, pada materi ini, kita hanya akan fokus pada bagian basic element dalam streamlit yang terdiri dari write, text, data display, dan chat. Yuk, kita bahas satu per satu!

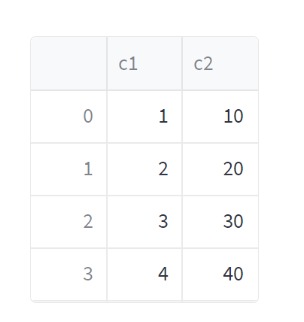
**Write**

Basic element pertama yang akan kita bahas ialah write. Ia merupakan elemen streamlit yang digunakan untuk menampilkan sebuah output.  Untuk menggunakan element ini, kita hanya perlu memanggil function write() dan diikuti inputan yang ingin ditampilkan.

Pada contoh pembuatan “Hello, world!” app, kita telah menggunakan function write() untuk menampilkan output dari argument markdown.  Sebenarnya function ini dapat digunakan untuk menampilkan hal lain, seperti DataFrame, visualisasi data, dll. Berikut merupakan contoh penggunaannya untuk menampilkan DataFrame.

|  |
| --- |
| import streamlit as st  import pandas as pd    st.write(pd.DataFrame({  'c1': [1, 2, 3, 4],  'c2': [10, 20, 30, 40],  })) |

Kode di atas akan menghasilkan tampilan web app seperti berikut.



Untuk melihat contoh lain dalam penggunaan function write(), Anda dapat mengunjungi dokumentasi berikut: [st.write()](https://docs.streamlit.io/library/api-reference/write-magic/st.write" \t "_blank).

**Text**

Elemen lain yang ada dalam streamlit ialah text (dokumentasi: [text](https://docs.streamlit.io/library/api-reference/text)). Sesuai dengan namanya, ia merupakan elemen yang digunakan untuk menampilkan sebuah output berupa text. Elemen ini memiliki banyak function yang bisa digunakan sesuai kebutuhan. Berikut merupakan beberapa pilihan yang tersedia saat ini.

* **markdown()**  
  Function ini digunakan untuk menampilkan output dari argument markdown. Berikut merupakan contoh kode untuk menggunakannya.

|  |
| --- |
| import streamlit as st    st.markdown(  """  # My first app  Hello, para calon praktisi data masa depan!  """  ) |

* **title()**  
  Ini merupakan function yang digunakan untuk menampilkan teks dalam format judul. Kode yang dapat Anda gunakan untuk menerapkan function tersebut adalah seperti di bawah ini.

|  |
| --- |
| import streamlit as st    st.title('Belajar Analisis Data') |

* **header()**  
  Function ini digunakan untuk menampilkan output teks sebagai format header. Berikut contoh penulisan kode untuk menggunakannya.

|  |
| --- |
| import streamlit as st  st.header('Pengembangan Dashboard') |

* **subheader()**  
  Ini merupakan function yang digunakan untuk menampilkan output teks sebagai format subheader.

|  |
| --- |
| import streamlit as st  st.subheader('Pengembangan Dashboard') |

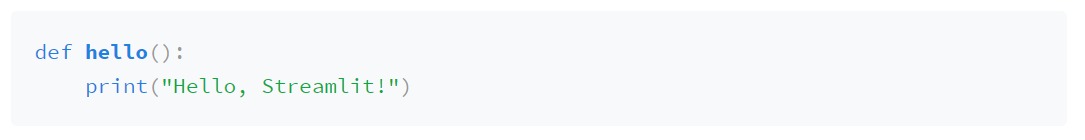
* **caption()**  
  Function berikutnya ialah caption(). Ia merupakan function yang digunakan untuk menampilkan output teks dalam ukuran kecil. Function ini biasanya digunakan untuk menampilkan caption, footnotes, dll. Contoh penggunaannya seperti di bawah ini.

|  |
| --- |
| import streamlit as st  st.caption('Copyright (c) 2023') |

* **code()**  
  Pada beberapa case, kita harus menampilkan potongan kode ke dalam streamlit app (web app yang dibuat menggunakan streamlit). Untuk menjawab hal ini, streamlit telah menyediakan sebuah function bernama code(). Kode di bawah ini merupakan contoh penggunaan dari function tersebut.

|  |
| --- |
| import streamlit as st    code = """def hello():  print("Hello, Streamlit!")"""  st.code(code, language='python') |

Kode di atas akan menghasilkan tampilan seperti berikut.



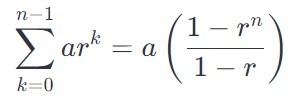
* **text()**  
  Function selanjutnya ialah text(). Function ini digunakan untuk menampilkan sebuah normal teks. Berikut merupakan contoh kode untuk mengguankannya.

|  |
| --- |
| import streamlit as st    st.text('Halo, calon praktisi data masa depan.') |

* **latex()**  
  Function terakhir yang dapat digunakan untuk menampilkan elemen teks ialah latex(). Sesuai namanya, function tersebut digunakan untuk menampilkan *mathematical expression* yang ditulis dalam [format LaTeX](https://katex.org/docs/supported.html). Berikut contoh kode untuk menggunakan function latex().

|  |
| --- |
| import streamlit as st    st.latex(r"""  \sum\_{k=0}^{n-1} ar^k =  a \left(\frac{1-r^{n}}{1-r}\right)  """) |

Kode tersebut akan menghasilkan tampilan mathematical expression seperti berikut.



**Data Display**

Basic element selanjutnya yang akan kita bahas ialah data display (dokumentasi: [data display](https://docs.streamlit.io/library/api-reference/data)). Ia merupakan elemen yang digunakan untuk menampilkan data secara cepat dan interaktif ke dalam streamlit app yang kita buat. Elemen ini memiliki beberapa function seperti berikut.

* **dataframe()**  
  Function pertama yang bisa kita gunakan untuk menampilkan data ke dalam streamlit app ialah dataframe(). Ia merupakan function yang digunakan untuk menampilkan DataFrame sebagai sebuah tabel interaktif. Pada function ini, kita bisa mengatur ukuran dari table yang ingin ditampilkan menggunakan parameter width dan height. Berikut merupakan contoh kode untuk menampilkan data menggunakan function dataframe().

|  |
| --- |
| import pandas as pd  import streamlit as st    df = pd.DataFrame({  'c1': [1, 2, 3, 4],  'c2': [10, 20, 30, 40],  })    st.dataframe(data=df, width=500, height=150) |

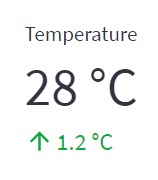
* **table()**  
  Selain dataframe(), kita juga bisa menggunakan function table()untuk menampilkan data ke dalam streamlit app. Ia dapat digunakan untuk menampilkan data dalam bentuk static table. Berikut merupakan contoh penggunaannya.

|  |
| --- |
| import pandas as pd  import streamlit as st    df = pd.DataFrame({  'c1': [1, 2, 3, 4],  'c2': [10, 20, 30, 40],  })  st.table(data=df) |

* **metric()**  
  Ketika membuat dashboard, terkadang kita perlu menampilkan sebuah metrik tertentu. Untuk melakukan hal ini, kita bisa memanfaatkan function metric(). Function ini dapat membantu kita untuk menampilkan sebuah metrik tertentu beserta detailnya seperti label, value serta besar perubahan nilainya. Berikut merupakan contoh kode untuk menggunakannya.

|  |
| --- |
| import streamlit as st    st.metric(label="Temperature", value="28 °C", delta="1.2 °C") |

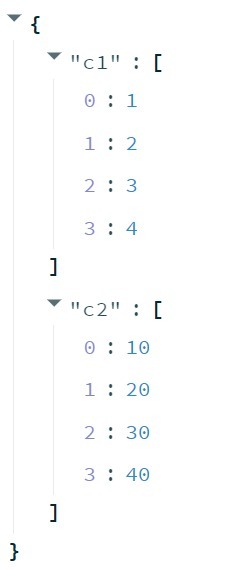
Kode di atas akan menghasilkan tampilan berikut.



* **json()**  
  Selain bentuk DataFrame atau tabel, terkadang kita juga perlu menampilkan data dalam format JSON. Streamlit telah menyediakan function json() untuk menampilkan data dalam format JSON. Berikut merupakan contoh penggunaannya.

|  |
| --- |
| import streamlit as st    st.json({  'c1': [1, 2, 3, 4],  'c2': [10, 20, 30, 40],  }) |

Kode di atas akan menghasilkan tampilan berikut.



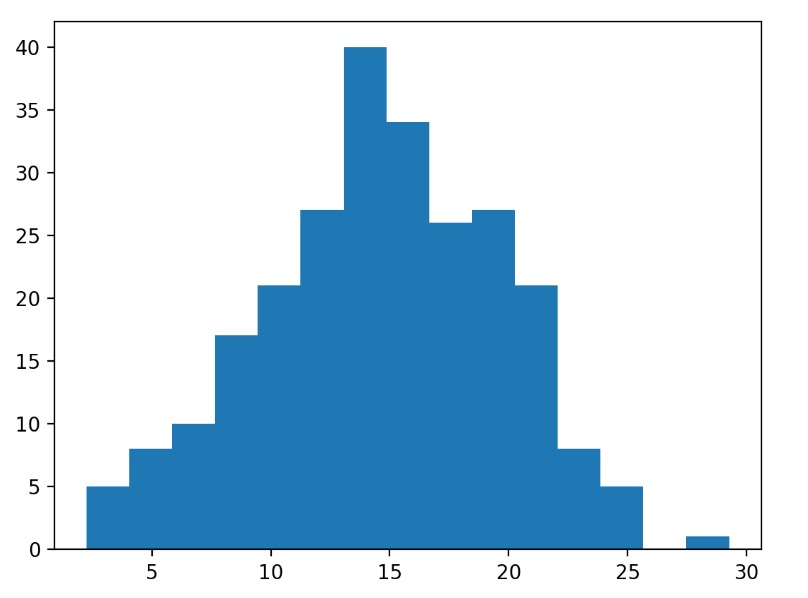
**Chart**

Basic element terakhir yang perlu kita ketahui ialah chart. Sesuai namanya, elemen ini dapat digunakan untuk menampilkan grafik visualisasi data ke dalam streamlit app. Elemen inilah yang akan sering kita gunakan untuk membuat dashboard nantinya. Sebenarnya streamlit telah menyediakan banyak sekali function untuk mendukung berbagai library visualisasi data (dokumentasi: [chart](https://docs.streamlit.io/library/api-reference/charts)). Namun, pada materi ini, kita hanya akan fokus pada function pyplot().

Function pyplot() dapat digunakan untuk menampilkan grafik visualisasi data yang dibuat menggunakan matplotlib. Berikut merupakan contoh kode untuk menggunakannya.

|  |
| --- |
| import matplotlib.pyplot as plt  import numpy as np  import streamlit as st    x = np.random.normal(15, 5, 250)    fig, ax = plt.subplots()  ax.hist(x=x, bins=15)  st.pyplot(fig) |

Kode di atas akan menampilkan grafik visualisasi data berikut di dalam streamlit app.



**Basic Widgets dalam Streamlit**

Kita telah mengenal berbagai basic element yang ada dalam streamlit. Hal tersebut tentunya akan sangat membantu kita dalam membuat dashboard. Namun, untuk menghasilkan sebuah dashboard yang interaktif, kita memerlukan komponen lain seperti *widget*(elemen *Graphical User Interface* yang memungkinkan pengguna untuk berinteraksi dengan aplikasi).

Nah, pada materi ini, kita akan berkenalan dengan berbagai basic widget yang telah disediakan oleh streamlit. Untuk membantu kita menghasilkan web app yang interaktif, streamlit telah menyediakan banyak sekali pilihan widget yang bisa disesuaikan dengan kebutuhan. Agar lebih mudah dalam memahami seluruh widget tersebut, pada materi ini kita akan membaginya ke dalam dua kategori, yaitu *input widget* dan *button widget*.

**Input Widget**

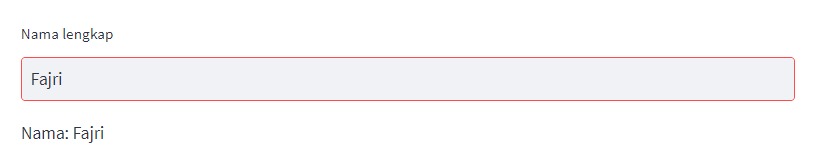
Kategori widget yang akan kita bahas ialah input widget. Ia merupakan kategori widget yang memungkinkan pengguna untuk memberikan input ke dalam streamlit app. Terdapat beberapa widget yang termasuk kategori ini, seperti *text input*, *number input*, *date input*, dll. Yuk, kita bahas satu per satu!

**Text input**

Text input merupakan widget yang digunakan untuk memperoleh inputan berupa *single-line text*. Kita bisa menggunakan function text\_input() untuk membuat widget ini. Berikut merupakan contoh kode untuk membuatnya.

|  |
| --- |
| import streamlit as st    name = st.text\_input(label='Nama lengkap', value='')  st.write('Nama: ', name) |

Kode di atas akan menghasilkan sebuah widget seperti berikut.

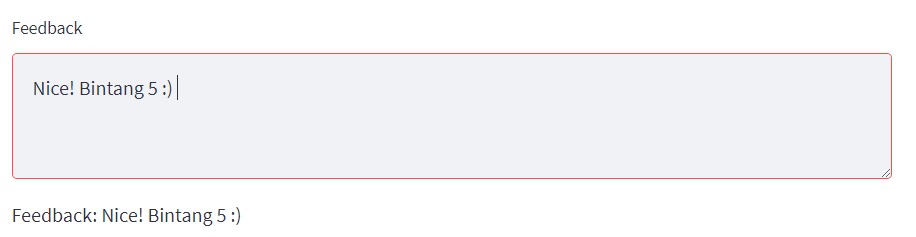


**Text-area**

Text area merupakan widget yang memungkinkan pengguna untuk menginput *multi-line text*. Untuk membuat widget ini, streamlit telah menyediakan function bernama text\_area(). Kode di bawah ini merupakan contoh kode untuk menggunakan function tersebut.

|  |
| --- |
| import streamlit as st    text = st.text\_area('Feedback')  st.write('Feedback: ', text) |

Berikut merupakan tampilan widget yang diperoleh dari kode di atas.

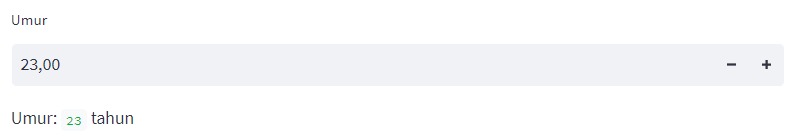


**Number input**

Widget berikutnya yang akan kita bahas ialah number input . Ia merupakan widget yang digunakan untuk memperoleh inputan berupa angka dari pengguna. Anda dapat menggunakan contoh kode berikut untuk membuat number input widget menggunakan function number\_input().

|  |
| --- |
| import streamlit as st    number = st.number\_input(label='Umur')  st.write('Umur: ', int(number), ' tahun') |

Gambar di bawah ini merupakan tampilan widget yang dihasilkan dari contoh kode tersebut.

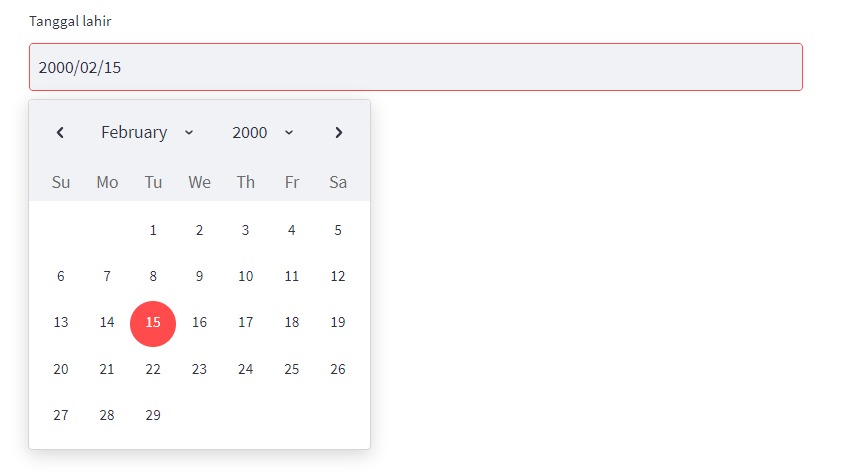


**Date input**

Selain inputan berupa angka dan teks, pada beberapa kasus kita juga membutuhkan input berupa tanggal dari pengguna melalui date input widget. Kita dapat membuat widget tersebut menggunakan function date\_input(). Berikut merupakan contoh kode untuk menggunakannya.

|  |
| --- |
| import datetime  import streamlit as st    date = st.date\_input(label='Tanggal lahir', min\_value=datetime.date(1900, 1, 1))  st.write('Tanggal lahir:', date) |

Kode tersebut akan menghasilkan tampilan widget seperti berikut.

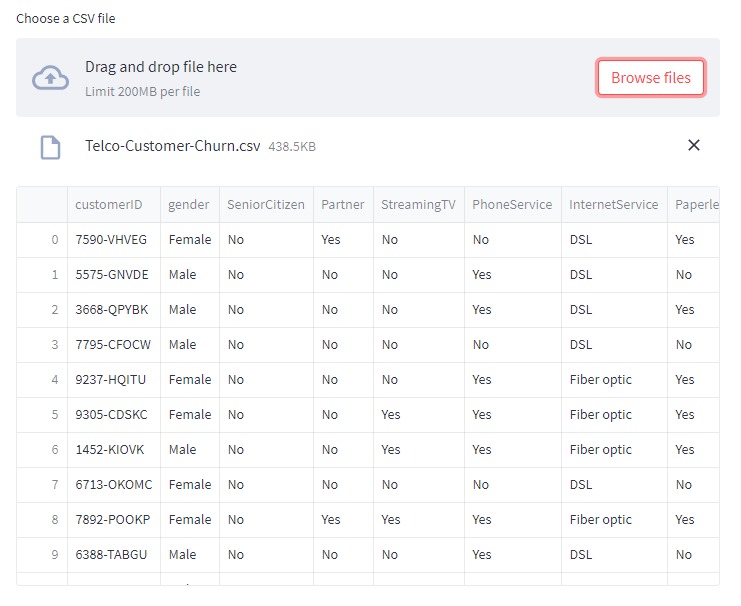


**File uploader**

Widget selanjutnya yang akan kita bahas ialah file uploader. Widget ini memungkinkan kita meminta pengguna untuk meng-*upload* sebuah berkas tertentu ke dalam web app. Kita dapat membuat file uploader widget menggunakan function file\_uploader() seperti pada contoh kode berikut.

|  |
| --- |
| import streamlit as st  import pandas as pd    uploaded\_file = st.file\_uploader('Choose a CSV file')    if uploaded\_file:  df = pd.read\_csv(uploaded\_file)  st.dataframe(df) |

Kode di atas akan menghasilkan widget seperti berikut.

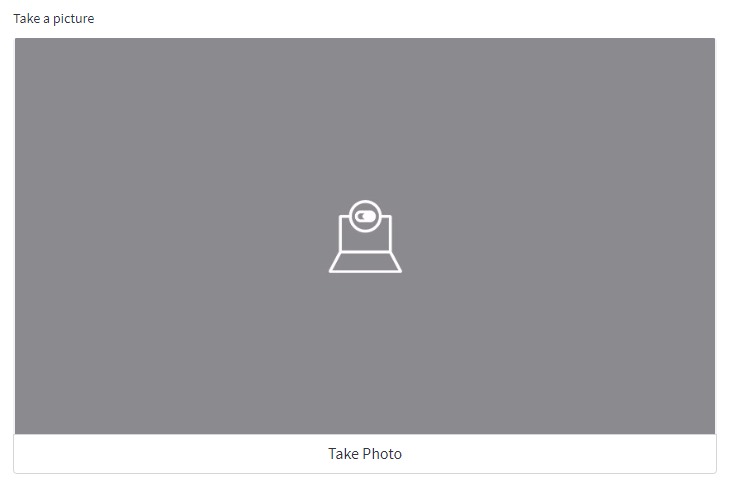


**Camera input**

Selain file uploader, streamlit juga menyediakan camera input widget yang dapat digunakan untuk meminta user mengambil gambar melalui *webcam* sekaligus mengunggahnya.  Hal ini tentunya dilakukan dengan persetujuan pengguna. Berikut merupakan contoh kode untuk membuat camera input widget menggunakan function camera\_input().

|  |
| --- |
| import streamlit as st  picture = st.camera\_input('Take a picture')  if picture:  st.image(picture) |

Berikut merupakan tampilan widget yang akan diperoleh ketika menjalankan kode di atas.



**Button Widgets**

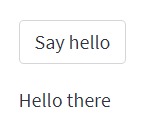
Oke, kategori widget selanjutnya yang akan kita bahas ialah button widget. Ia merupakan kategori widget yang terdiri dari button, checkbox, radio button, dll.

**Button**

Button merupakan widget untuk menampilkan tombol interaktif. Tombol tersebut dapat digunakan pengguna untuk melakukan aksi tertentu. Untuk membuat widget ini, kita bisa menggunakan function button() seperti contoh berikut.

|  |
| --- |
| import streamlit as st    if st.button('Say hello'):  st.write('Hello there') |

Berikut merupakan button widget yang dihasilkan dari kode di atas.

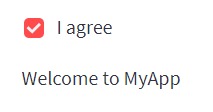


**Checkbox**

Checkbox merupakan widget yang digunakan untuk menampilkan sebuah *checklist* untuk pengguna. Kita bisa menggunakan function checkbox() untuk membuat dan menampilkan checklist dalam streamlit app. Berikut contoh kode untuk melakukannya.

|  |
| --- |
| import streamlit as st    agree = st.checkbox('I agree')    if agree:  st.write('Welcome to MyApp') |

Kode di atas akan menghasilkan tampilan berikut.

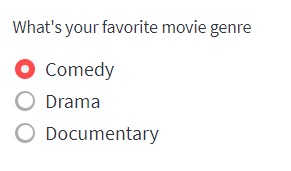


**Radio button**

Selain button dan checkbox, terkadang kita juga membutuhkan radio button untuk menghasilkan web app yang interaktif. Ia merupakan widget yang memungkinkan pengguna untuk memilih satu dari beberapa pilihan yang ada. Untuk membuat widget ini, kita bisa menggunakan function radio() seperti pada contoh kode berikut.

|  |
| --- |
| import streamlit as st    genre = st.radio(  label="What's your favorite movie genre",  options=('Comedy', 'Drama', 'Documentary'),  horizontal=False  ) |

Berikut merupakan tampilan widget yang dihasilkan dari kode di atas.

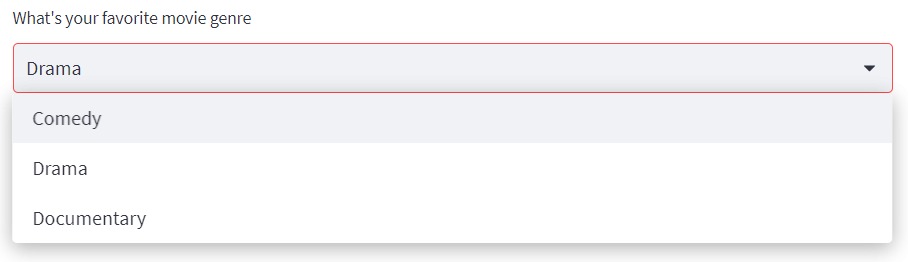


**Select Box**

Select box merupakan widget yang memungkinkan pengguna untuk memilih salah satu dari beberapa pilihan yang ada. Ia merupakan opsi alternatif dari radio button. Streamlit telah menyediakan function selectbox() untuk membuat select box widget. Berikut contoh penggunaannya.

|  |
| --- |
| import streamlit as st    genre = st.selectbox(  label="What's your favorite movie genre",  options=('Comedy', 'Drama', 'Documentary')  ) |

Berikut merupakan tampilan select box dari kode di atas.

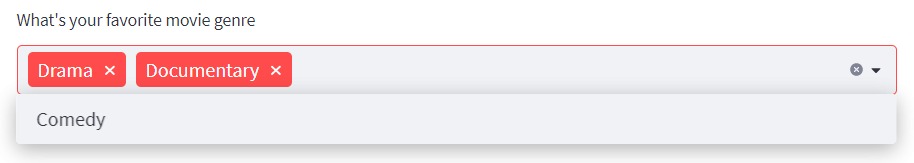


**Multiselect**

Widget lain yang harus kita ketahui ialah multiselect. Ia merupakan widget yang digunakan agar user dapat memilih lebih dari satu pilihan dari sekumpulan pilihan yang ada. Untuk mempermudah kita dalam membuat multiselect widget, streamlit telah menyediakan function bernama multiselect(). Berikut contoh kode untuk menggunakan function tersebut.

|  |
| --- |
| import streamlit as st    genre = st.multiselect(  label="What's your favorite movie genre",  options=('Comedy', 'Drama', 'Documentary')  ) |

Kode di atas akan menghasilkan tampilan multiselect widget seperti berikut.

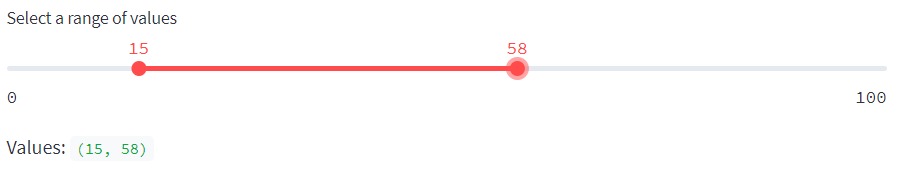


**Slider**

Slider merupakan widget yang memungkinkan pengguna untuk untuk memilih nilai (atau range nilai) dari sebuah range nilai yang telah ditentukan. Streamlit telah menyediakan function slider() untuk membuat slider widget. Berikut merupakan contoh penggunaannya.

|  |
| --- |
| values = st.slider(  label='Select a range of values',  min\_value=0, max\_value=100, value=(0, 100))  st.write('Values:', values) |

Kode di atas akan menghasilkan widget seperti berikut.



Nah, itulah beberapa pilihan widget yang disediakan oleh streamlit. Anda dapat melihat lebih banyak pilihan widget pada dokumentasi berikut: [widgets in streamlit](https://docs.streamlit.io/library/api-reference/widgets). Pengetahuan terkait basic widget ini akan sangat membantu Anda dalam membuat dashboard yang interaktif nantinya.

Oke, sebelum kita belajar membuat dashboard, terdapat satu materi lagi yang harus dipahami, yaitu terkait layout dalam streamlit. *So*, tetap semangat, ya!

**Basic Layouts dalam Streamlit**

Sejauh ini Anda telah mengenal berbagai hal mulai dari basic element hingga beberapa pilihan widget yang ada dalam streamlit. Semua hal tersebut tentunya akan sangat membantu kita dalam membuat sebuah dashboard yang interaktif.

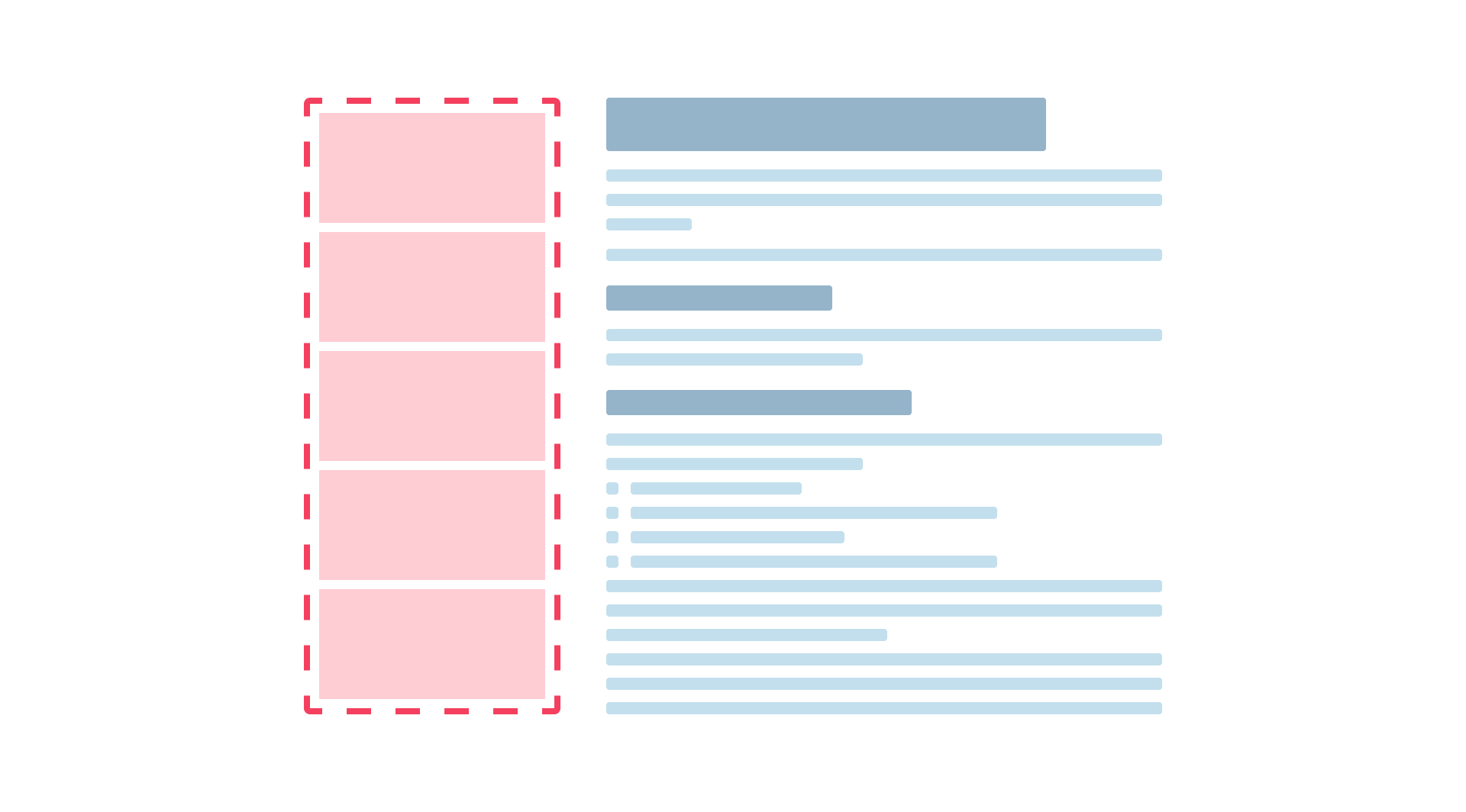
Apakah itu sudah cukup? Tentu saja belum!

Untuk membuat dashboard yang rapi, kita perlu belajar cara mengatur layout atau tampilan dari sebuah dashboard. Nah, pada materi kali ini, kita akan mengupas tuntas tentang basic layout dalam streamlit.

Sebagai sebuah web app framework, streamlit telah menyediakan berbagai pilihan layout yang dapat digunakan untuk mengatur tampilan web app (atau dashboard) yang akan dibuat. Pilihan layout yang tersedia antara lain *sidebar*, *columns*, *tabs*, *expander*, serta *container*. Tentunya setiap pilihan layout tersebut memiliki fungsi dan kegunaannya masing-masing. Yuk, kita bahas satu per satu!

**Sidebar**

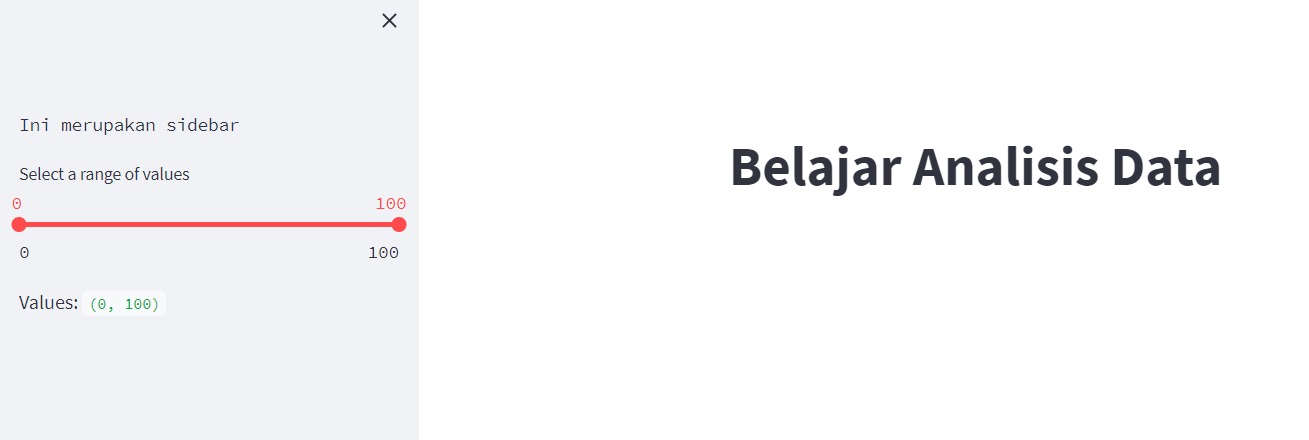
Elemen layout pertama yang akan kita bahas ialah sidebar. Ia merupakan area yang berada di samping konten utama. Pada streamlit, posisi sidebar secara default berada di bagian kiri dari konten utama dan dapat memuat berbagai hal mulai dari gambar, teks, hingga widget.



Untuk menambahkan sebuah elemen atau widget ke dalam sidebar, kita bisa menggunakan notasi with yang diikuti sebuah object bernama sidebar yang telah disediakan oleh streamlit. Berikut merupakan contoh kode untuk menambah sebuah elemen dan widget ke dalam sidebar.

|  |
| --- |
| import streamlit as st    st.title('Belajar Analisis Data')    with st.sidebar:    st.text('Ini merupakan sidebar')    values = st.slider(  label='Select a range of values',  min\_value=0, max\_value=100, value=(0, 100)  )  st.write('Values:', values) |

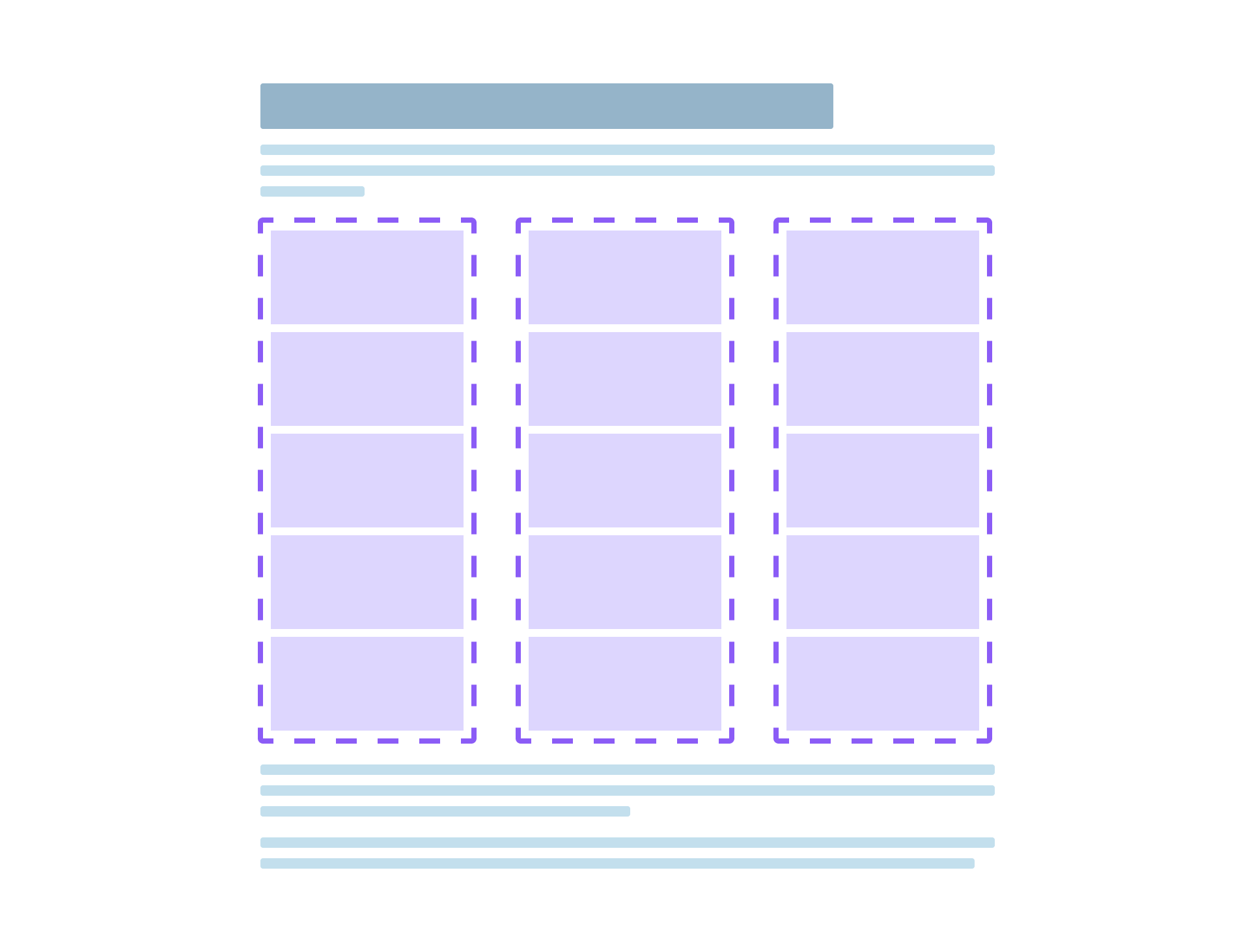
Kode di atas akan menghasilkan tampilan streamlit app seperti berikut.



Bagaimana cukup mudah bukan untuk membuat sidebar di streamlit? (dokumentasi: [sidebar](https://docs.streamlit.io/library/api-reference/layout/st.sidebar)). Nah, sidebar ini dapat kita gunakan untuk menampung gambar logo serta widget yang digunakan sebagai filter. Namun, hal ini akan kita bahas pada materi berikutnya.

**Columns**

Columns merupakan elemen layout yang memungkinkan kita untuk mengatur tampilan pada konten utama ke dalam beberapa kolom sesuai kebutuhan. Gambar berikut merupakan ilustrasi tampilan dari elemen layout ini.



Untuk membuat column dalam streamlit app, kita harus membuat object dari setiap kolom terlebih dahulu. Hal ini dapat dilakukan dengan memanfaatkan function columns(). Selanjutnya, kita hanya perlu menambahkan sebuah elemen atau widget ke dalam column tersebut menggunakan notasi with. Berikut merupakan contoh kode untuk membuat column dalam streamlit.

|  |
| --- |
| import streamlit as st    st.title('Belajar Analisis Data')  col1, col2, col3 = st.columns(3)    with col1:  st.header("Kolom 1")  st.image("https://static.streamlit.io/examples/cat.jpg")    with col2:  st.header("Kolom 2")  st.image("https://static.streamlit.io/examples/dog.jpg")    with col3:  st.header("Kolom 3")  st.image("https://static.streamlit.io/examples/owl.jpg") |

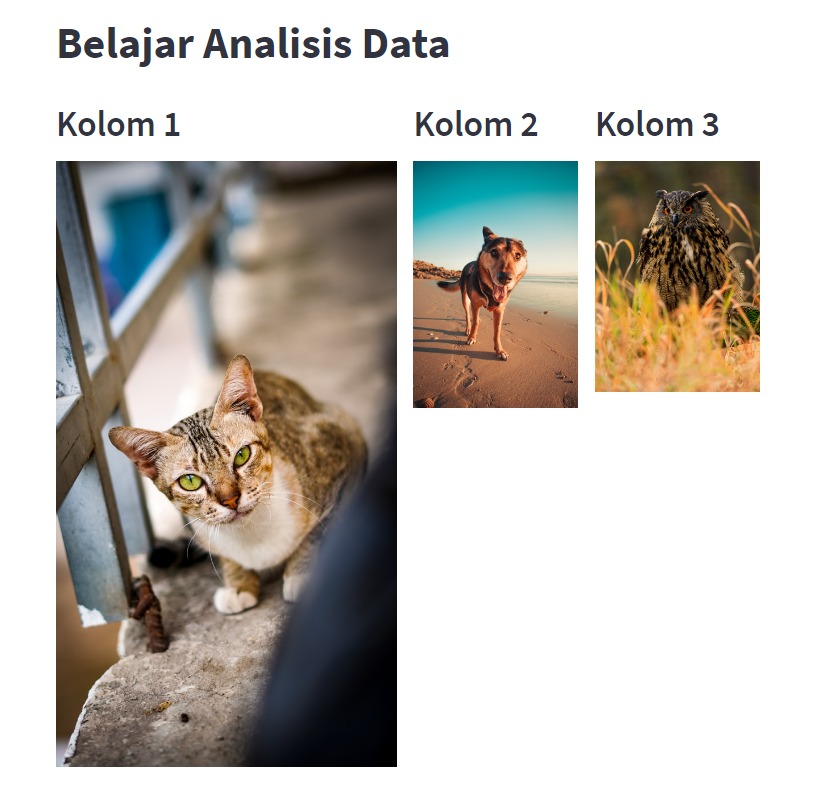
Berikut tampilan streamlit app yang dihasilkan dari kode di atas.



Sebenarnya, kita bisa dengan bebas mengatur ukuran dari column yang dibuat. Nah, untuk melakukan hal ini, kita harus memasukkan sebuah*list* yang berisi perbandingan ukuran dari kolom yang akan dibuat. Sebagai contoh, kode di bawah ini akan menampilkan 3 buah kolom  dengan perbandingan**2:1:1**.

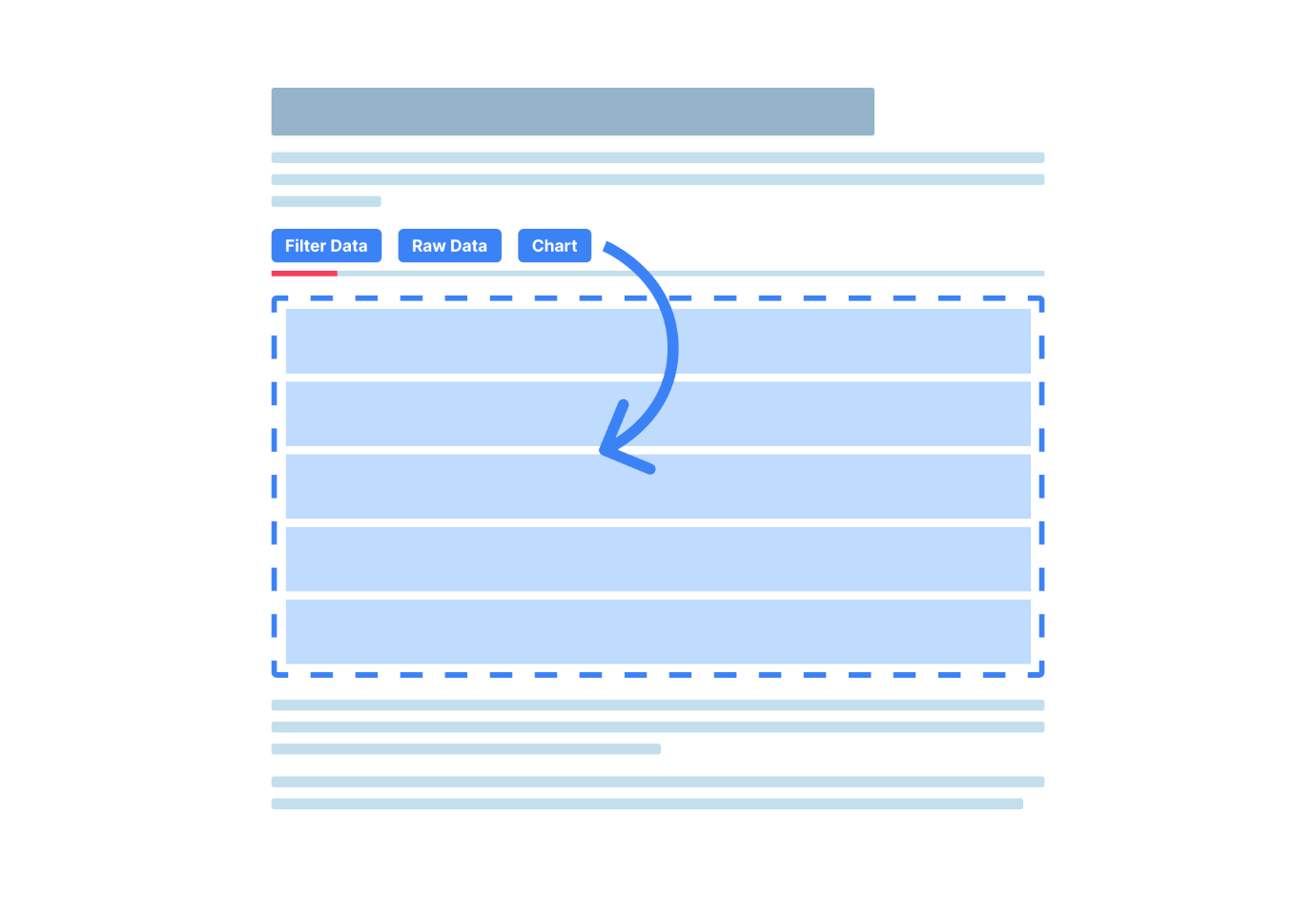
|  |
| --- |
| import streamlit as st    st.title('Belajar Analisis Data')  col1, col2, col3 = st.columns([2, 1, 1])    with col1:  st.header("Kolom 1")  st.image("https://static.streamlit.io/examples/cat.jpg")    with col2:  st.header("Kolom 2")  st.image("https://static.streamlit.io/examples/dog.jpg")    with col3:  st.header("Kolom 3")  st.image("https://static.streamlit.io/examples/owl.jpg") |

Berikut merupakan tampilan yang akan dihasilkan dari kode tersebut.



**Tabs**

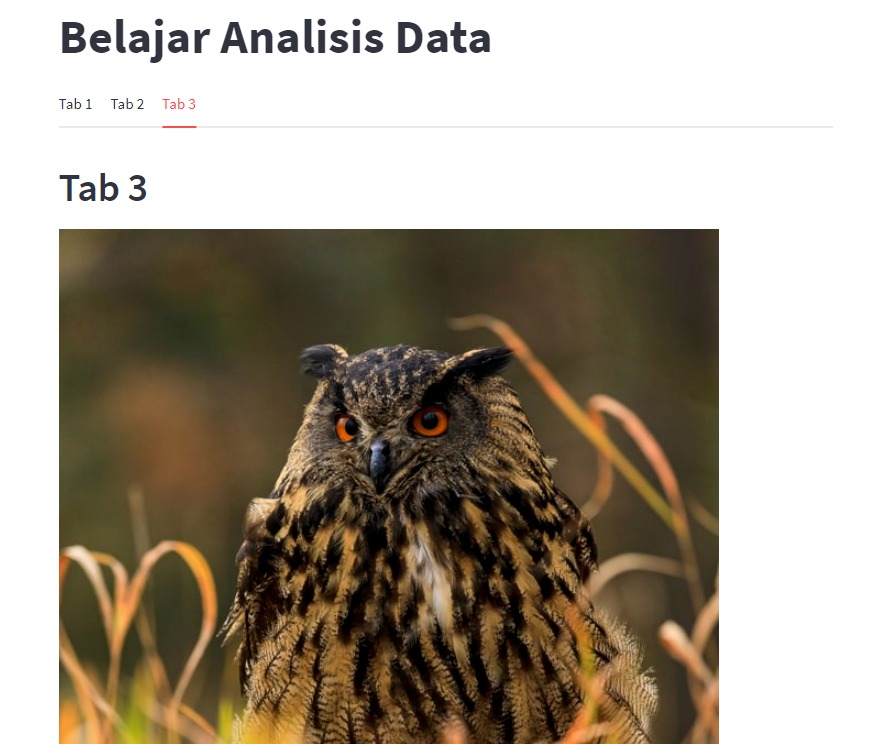
Elemen layout berikutnya yang terdapat dalam streamlit ialah tabs. Ia merupakan elemen layout yang memungkinkan kita untuk menambahkan beberapa tabs ke dalam konten utama. Hal ini tentunya akan sangat membantu kita dalam menghasilkan dashboard yang interaktif.



Sama halnya dengan columns yang sebelumnya kita bahas, untuk membuat tabs, kita harus membuat object dari setiap tab menggunakan function tabs() yang diikuti dengan list dari nama dari setiap tab. Berikut contoh kode untuk membuat tabs dalam streamlit app.

|  |
| --- |
| import streamlit as st    st.title('Belajar Analisis Data')  tab1, tab2, tab3 = st.tabs(["Tab 1", "Tab 2", "Tab 3"])    with tab1:  st.header("Tab 1")  st.image("https://static.streamlit.io/examples/cat.jpg")    with tab2:  st.header("Tab 2")  st.image("https://static.streamlit.io/examples/dog.jpg")    with tab3:  st.header("Tab 3")  st.image("https://static.streamlit.io/examples/owl.jpg") |

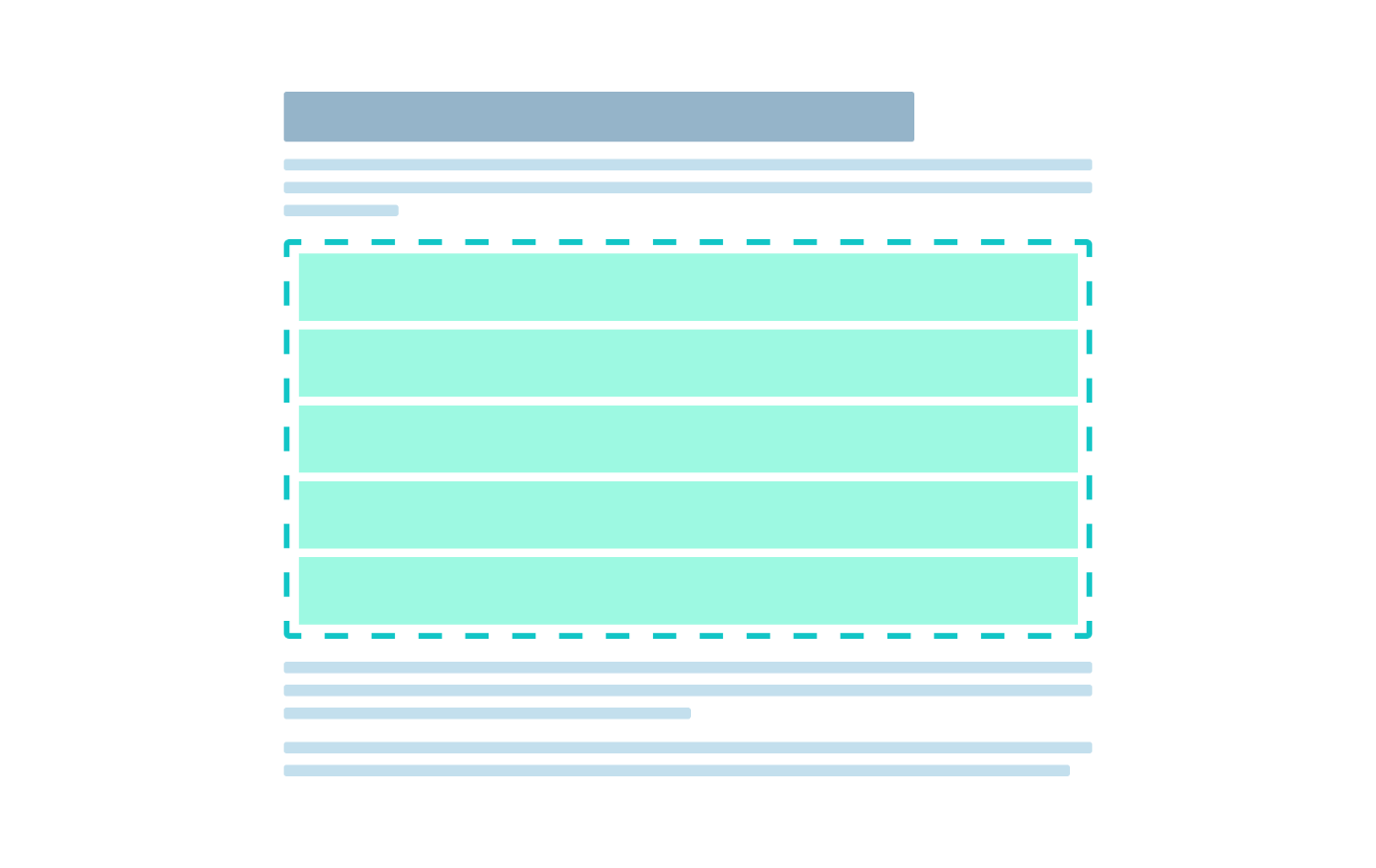
Berikut merupakan tampilan tabs yang akan Anda peroleh dari kode di atas.



Nah, itulah cara membuat tabs dalam streamlit app (dokumentasi: [tabs](https://docs.streamlit.io/library/api-reference/layout/st.tabs)). Cukup mudah, bukan?

**Container**

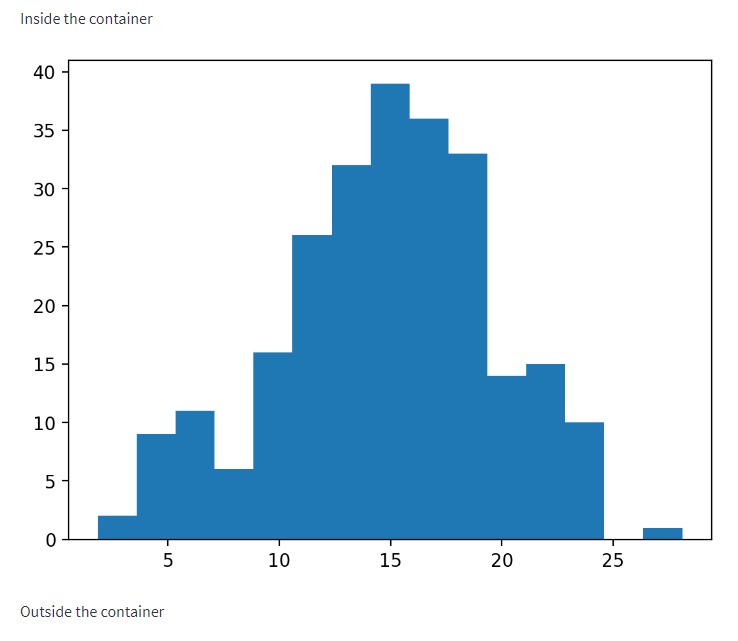
Container merupakan elemen layout dalam streamlit yang memungkinkan kita untuk membuat sebuah wadah untuk menampung suatu atau beberapa elemen dengan ukuran yang tetap. Container ini dapat kita gunakan untuk menghasilkan dashboard yang rapi.



Untuk membuat container, kita bisa menggunakan notasi with dan diikuti function container(). Kode di bawah ini merupakan contoh kode untuk membuat container dalam streamlit app.

|  |
| --- |
| import streamlit as st  import matplotlib.pyplot as plt  import numpy as np    with st.container():  st.write("Inside the container")    x = np.random.normal(15, 5, 250)    fig, ax = plt.subplots()  ax.hist(x=x, bins=15)  st.pyplot(fig)    st.write("Outside the container") |

Berikut merupakan tampilan streamlit app dari kode di atas.



Ketika melihat gambar di atas, mungkin Anda tidak melihat adanya perbedaan dari tampilan sebelum dan sesudah menggunakan container. Hal ini terjadi karena kita masih belum memiliki banyak elemen yang ingin ditampilkan.

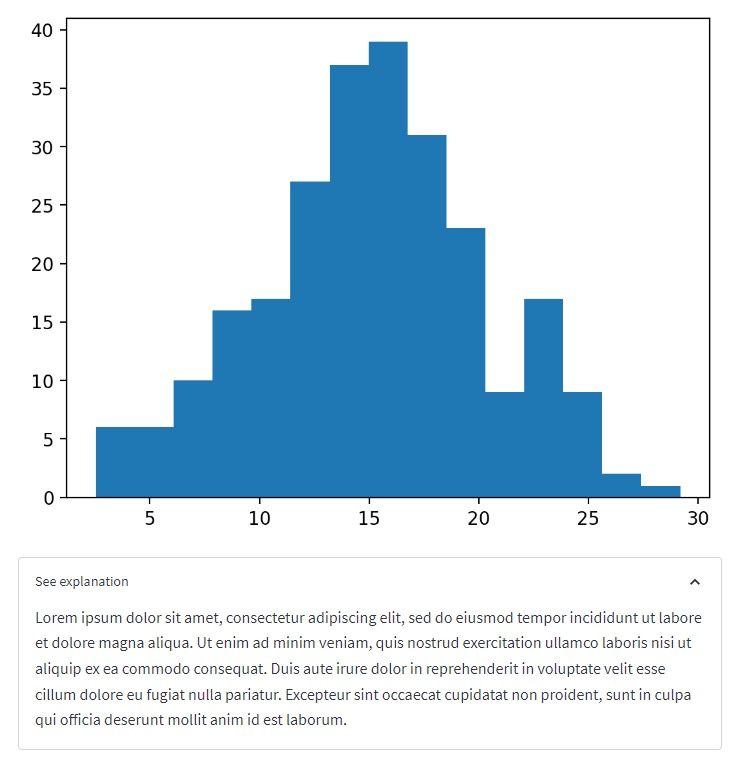
**Expander**

Elemen layout selanjutnya yang tidak kalah penting ialah expander. Anda dapat menganggap elemen layout ini sebagai sebuah container yang dapat diperluas atau diciutkan secara interaktif.



Untuk membuat elemen layout ini, kita bisa menggunakan notasi with dan diikuti dengan function expander() seperti pada contoh kode berikut.

|  |
| --- |
| import streamlit as st  import matplotlib.pyplot as plt  import numpy as np    x = np.random.normal(15, 5, 250)    fig, ax = plt.subplots()  ax.hist(x=x, bins=15)  st.pyplot(fig)    with st.expander("See explanation"):  st.write(  """Lorem ipsum dolor sit amet, consectetur adipiscing elit,  sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.  Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris  nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor  in reprehenderit in voluptate velit esse cillum dolore eu fugiat  nulla pariatur. Excepteur sint occaecat cupidatat non proident,  sunt in culpa qui officia deserunt mollit anim id est laborum.  """  ) |



Seperti yang terdapat pada gambar di atas, kita bisa menggunakan expander untuk menampung penjelasan atau keterangan lanjutan dari sebuah visualisasi data yang ditampilkan dalam sebuah dashboard.

Nah, itulah beberapa pilihan layout yang disediakan oleh streamlit untuk membantu kita dalam membuat web app yang rapi dan interaktif. Apabila ingin mempelajari topik ini lebih lanjut, Anda dapat mengunjungi dokumentasi berikut: [layouts and containers](https://docs.streamlit.io/library/api-reference/layout).