Chapter 4: Data Elements

- Hi! So, in the last chapter, we learnt much about Streamlit's Text Elements. Now, we will also learn how to utilise the coolest parts of Streamlit, the Data Elements.

- Data is information such as facts, numbers, figures etc. which we can manipulate. We can represent it in clean ways. Streamlit's Data Elements provide visually stunning ways to show data.

- Here's a list of all the native components inside the Data Elements category.

No description has been provided for this image

- So, let's get started on learning about these Data Elements.

`st.dataframe`

- Displays a dataframe as an interactive table. This improves the UX and makes it cool.

- You can even hover to the top-right corner of the table and get options to download the table as a `CSV/Comma-Seperated Values` worksheet, search for values and view fullscreen.

- It is best used with certain DataFrame-like objects of some libraries such as `pandas`, `polars` and `snowflake`. The most common example is the well-used `pandas.DataFrame` class.

- You can pass the DataFrame object in the `data` parameter. Streamlit will automatically apply and render it.

- The function signature is: st.dataframe(`data` = `None`, `width` =

`"stretch"`. `height` = `"auto"`, `use_container_width` = `None`, `hide_index` = `None`, `column_order` = `None`, `column_config` = `None`, `key` = `None`, `on_select` = `"ignore"`, `selection_mode` = `"multi-row"`, `row_height` = `None`)

- Among these, `use_container_width` is deprecated. It is generally best to avoid using it. An alternative if you really have to set `use_container_width` to `True` is `width` = `"stretch"`.

- An example is:

In [ ]:
```
import pandas as pd
import streamlit as st
from numpy.random import default_rng as rng

df = pd.DataFrame(
    rng(0).standard_normal((8, 4)), columns=("col %d" % i for i in range(4))
) # This DataFrame generates random numbers. There are 8 rows and 3 columns.

st.dataframe(df)
```

No description has been provided for this image

- Note: This topic will get too much advanced. As this book targets simplicity, not all sorts of examples will be covered. So, you can explore more of some examples in here.

`st.data_editor`

- Displays a data editor widget. As we explored before, the `st.dataframe` displays a DataFrame in a tabular form. But the `st.data_editor` widget allows us to create a similar table, but with the ability to modify data. You can do this by double-clicking on a cell, editing its contents and clicking anywhere outside the modified cell.

- The funtction signature is: st.data_editor(`data`, `width` = `"stretch"`, `height` = `"auto"`, `use_container_width` = `None`, `hide_index` = `None`, `column_order` = `None`, `column_config` = `None`, `num_rows` = `"fixed"`, `disabled` = `False`, `key` = `None`, `on_change` = `None`, `args` = `None`, `kwargs` = `None`, `row_height` = `None`)

- One of the most basic example with live updating is:

In [ ]:
```python
import pandas as pd
import streamlit as st

df = pd.DataFrame(
    [
        {"command": "st.selectbox", "rating": 4, "is_widget": True},
        {"command": "st.balloons", "rating": 5, "is_widget": False},
        {"command": "st.time_input", "rating": 3, "is_widget": True},
    ]
) # Our DataFrame

edited_df = st.data_editor(df) # Load the `df` here

# This changes live
favorite_command = edited_df.loc[edited_df["rating"].idxmax()]["command"]
# Accordingly so does this
st.markdown(f"Your favorite command is **{favorite_command}** 🎈")
```

No description has been provided for this image