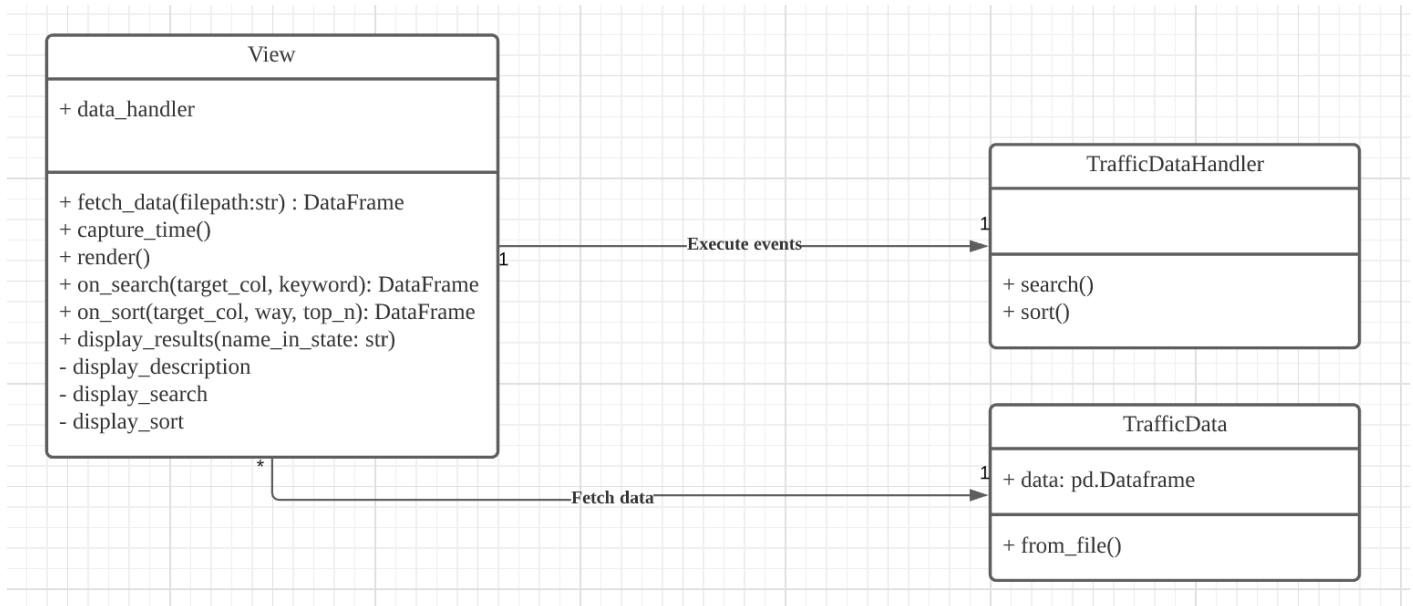


DDM5051 Project 2

- Group 8
- WONG Ryan, ZHAO Jiawei, CHEN Xinyi
- Version: 1.0, date: 06/12/2021

1. UML design



1.1 View

The main function of view object is to display the user interface. We use the 'streamlit' library which is easy to build a web and share data in Python.

	Name	Function
Attributes	data_handler	process data
Operations	fetch_data	read data from local path
	capture_time()	return search/sort time consuming
	render()	display user interface
	on_search(target_col, keyword)	check input
	on_sort(target_col, way, top_n)	check input
	display_results(name_in_state: str)	display pagination of large amount of results on web
	display_description	display column description on web
	display_search	disply search interface and search results on web
	display_sort	disply sort interface and sort results on web

1.2 TrafficData

The main function of this part is to read data from .csv and return it in pandas.dataframe format.

	Name	Function
Attributes	data:pandas.dataframe	None
Operations	from_file()	read .csv from local path and return a dataframe



1.3 TrafficDataHandler

This part is for sorting and searching. We use the sort and search functions in pandas.

	Name	Function
Attributes	None	None
Operations	search()	return lines containing the keyword
	sort()	sort the chosen column, return the sorted dataframe

2. Interface design

The whole interface designing progress is rely on the Python open source library – streamlit (st for short)

 **Inquiry System For Taiwan Traffic Data** 

Expand to preview the data. +

SEARCH

Search a record

keyword

VehicleType

Input your keyword

Search 🔍

SORT

Sort a record

Ascending order or not

Max. number of items

VehicleType

ascending

default: 10

Sort 🔍

2.1 Title

We use `st.header` to display the title.

```
st.header('🚦Inquiry System For Taiwan Traffic Data🚦')
```

2.2 Data preview

This part displays a small fraction of dataset and the column descriptions in order to introduce the data structure to users.

Inquiry System For Taiwan Traffic Data

Expand to peek the data and columns description.

First 5 rows of the dataset

	VehicleType	DerectionTime_O	GantryID_O	DerectionTime_D	GantryID_D
0	31	2019-08-30 08:17:14	03F3307N	2019-08-30 08:17:14	03F3307N
1	31	2019-08-30 08:01:02	03F0648S	2019-08-30 08:01:02	03F0648S
2	31	2019-08-30 08:51:45	03F0648N	2019-08-30 08:57:09	03F0559N
3	31	2019-08-30 08:20:45	03F3854S	2019-08-30 08:20:45	03F3854S
4	32	2019-08-30 08:06:10	03F4168S	2019-08-30 08:09:33	03F4232S

Column descriptions:

Columns	Descriptions
'VehicleType'	車種, 31小客車、32小貨車、41大客車、42大貨車、5聯結車
'DerectionTime_O'	車輛通過本旅次第1個測站時間
'GantryID_D'	車輛通過本旅次第1個測站編號
'DerectionTime_D'	車輛通過本旅次最後1個測站時間
'TripLength'	本旅次行駛距離
'TripEnd'	旅次標記(Y正常結束, N異常)
'TripInformation'	本旅次經過各個測站之通過時間及編號

```
def render(self):
    ##### loading the data
    with st.expander('Expand to peek the data and columns description.'):
        self._display_data_description()
def _display_data_description(self):
    st.markdown("#### First 5 rows of the dataset")
    st.write(st.session_state.traffic_df.head())
    st.markdown("""
    #### Column descriptions:
    | Columns | Descriptions |
    |---|---|
    | 'VehicleType' | 車種, 31小客車、32小貨車、41大客車、42大貨車、5聯結車 |
    | 'DerectionTime_O' | 車輛通過本旅次第1個測站時間 |
    | 'GantryID_D' | 車輛通過本旅次第1個測站編號 |
    | 'DerectionTime_D' | 車輛通過本旅次最後1個測站時間 |
    | 'TripLength' | 本旅次行駛距離 |
    | 'TripEnd' | 旅次標記(Y正常結束, N異常) |
    | 'TripInformation' | 本旅次經過各個測站之通過時間及編號 |

    """)
```

2.3 Search part

In search part, users will:

1. choose a column which contains the searching target
2. input the searching keyword

And our server will show all lines contain the keyword.

```
def _display_search(self):
    st.markdown("### SEARCH")
    search_spaces1, search_spaces2 = st.columns([1, 1])
```

```

search_target = search_spaces1.selectbox(
    "Search a record", st.session_state.traffic_df.columns)
is_search = search_spaces1.button("Search 🔍", )
if search_target in ['DerectionTime_0', 'DerectionTime_D']:
    search_keyword = search_spaces2.text_input(
        " ", placeholder="(YYYY-MM-DD HH:MM:SS)"
    )
else:
    search_keyword = search_spaces2.text_input(
        "keyword", placeholder="Input your keyword"
    )

if is_search:
    with st.spinner(f'Searching: {search_keyword} ...'):
        result = self.on_search(search_target, search_keyword)
        st.session_state.search_result = result
if st.session_state.get('search_result') is not None:
    st.subheader("Search results:")
    self.display_results('search_result')

```

2.4 Sort part

The appearance of sort part is similar to the search one. Here users will:

1. choose a column which is expected to be sorted
2. choose the sort way (ascending or non-ascending)
3. input the Max. number of items (for example, show the top3 largest items)

And our server will return a sorted dataframe.

```

def _display_sort(self):
    st.markdown("### SORT")
    sort_spaces1, sort_spaces2, sort_spaces3 = st.columns(3)

    sort_target = sort_spaces1.selectbox(
        "Sort a record", st.session_state.traffic_df.columns, key = "<aaa>")
    sort_way = sort_spaces2.selectbox(
        "Ascending order or not", ['ascending', 'descending'])
    sort_display_num = sort_spaces3.text_input(
        "Max. number of items", placeholder="default: 10")
    is_sort = sort_spaces1.button("Sort 🔍")

    if is_sort:
        result = self.on_sort(sort_target, sort_way, sort_display_num)
        st.session_state.sort_result = result
    if st.session_state.get('sort_result') is not None:
        st.subheader("Sort results:")
        self.display_results('sort_result')

```

3. Function design

Function part contains search and sort, which are realized based on Pandas.

To be noticed, when users do searching, firstly our function will check every element which is equal to the keyword strictly. If there is no result, it will check the elements which contain the keyword. For example, if you search the keyword '2019-08', all dates in Aug. 2019 will be returned.

```

class TrafficDataHandler:
    def search(self, df, column, keyword):
        r = df[df[column].isin([keyword])]
        if r.empty:
            return df[df[column].astype(str).str.contains(keyword)]
        return r

    def sort(self, df, column, way, num):
        if way == 'ascending':
            acd = True
        elif way == 'non-ascending':
            acd = False
        return df.sort_values(by=[column], ascending=acd)[:num]

```

4. Run example

4.1 Search

1. when user inputs the legal information, the result will return in a dataframe.

- search a date

SEARCH

Search a record

DerectionTime_O ▼ 2019-08-30

Search

Search done in 0.3857s

Search results:

Pagination
Page 1 of 25963

Page 1 of 25963

	VehicleType	DerectionTime_O	GantryID_O	DerectionTime_D	GantryID_D
0	31	2019-08-30 08:17:14	03F3307N	2019-08-30 08:17:14	03F3307N
1	31	2019-08-30 08:01:02	03F0648S	2019-08-30 08:01:02	03F0648S
2	31	2019-08-30 08:51:45	03F0648N	2019-08-30 08:57:09	03F0559N
3	31	2019-08-30 08:20:45	03F3854S	2019-08-30 08:20:45	03F3854S
4	32	2019-08-30 08:06:10	03F4168S	2019-08-30 08:09:33	03F4232S
5	31	2019-08-30 08:52:35	03F4168S	2019-08-30 08:58:14	03F4263S
6	32	2019-08-30 08:50:44	01F0681N	2019-08-30 08:51:54	01F0664N
7	31	2019-08-30 08:09:16	01F3640S	2019-08-30 08:09:16	01F3640S
8	31	2019-08-30 08:18:47	01F3640N	2019-08-30 08:18:47	01F3640N
9	31	2019-08-30 08:30:45	01F3590S	2019-08-30 08:34:24	01F3640S

- search a gantryID

SEARCH

Search a record

keyword

GantryID_O

03F4168S

Search

Search done in 0.0309s

Search results:

Pagination

Page 1 of 88

Page 1 of 88

Page 88 of 88

	VehicleType	DerectionTime_O	GantryID_O	DerectionTime_D	GantryID_O
4	32	2019-08-30 08:06:10	03F4168S	2019-08-30 08:09:33	03F4232S
5	31	2019-08-30 08:52:35	03F4168S	2019-08-30 08:58:14	03F4263S
349	31	2019-08-30 08:37:09	03F4168S	2019-08-30 08:37:09	03F4168S
1667	32	2019-08-30 08:28:34	03F4168S	2019-08-30 08:34:14	03F4263S
1873	32	2019-08-30 08:38:56	03F4168S	2019-08-30 08:43:01	03F4232S
2375	31	2019-08-30 08:55:22	03F4168S	2019-08-30 09:00:36	03F4263S
2488	32	2019-08-30 08:26:04	03F4168S	2019-08-30 08:31:20	03F4263S
3182	5	2019-08-30 08:04:14	03F4168S	2019-08-30 08:08:34	03F4232S
3605	31	2019-08-30 08:15:11	03F4168S	2019-08-30 08:18:42	03F4232S
3902	31	2019-08-30 08:33:18	03F4168S	2019-08-30 08:37:14	03F4232S

2. when user inputs an empty string, warning info will be shown.

SEARCH

Search a record

keyword

GantryID_O

Input your keyword

Search

Search done in 0.0000s

Search results:

please enter a keyword

4.1 Sort

1. the top 3 longest trip length

SORT

Sort a record

Ascending order or not

Max. number of items

TripLength

descending

3

Sort

Done in 0.2225s

Sort results:

	DerectionTime_D	GantryID_D	TripLength	TripEnd	TripInformation
136512	2019-08-30 12:22:14	03F0140N	417.8000	Y	2019-08-30 08:22:42+03F4...
197346	2019-08-30 13:27:03	03F0054N	412.8000	Y	2019-08-30 08:48:58+03F4...
197772	2019-08-30 13:58:46	01F0061N	410.8000	Y	2019-08-30 08:52:17+03F4...

2. the top 100 smallest vehicle type

SORT

Sort a record

Ascending order or not

Max. number of items

VehicleType

ascending

100

Sort 🔍

Done in 0.0737s

Sort results:

Pagination

Page 1 of 11

Page 1 of 11

Page 11 of 11

	VehicleType	DerectionTime_O	GantryID_O	DerectionTime_D	GantryID
82767	5	2019-08-30 08:47:40	01F0182S	2019-08-30 08:56:04	01F0293
68470	5	2019-08-30 08:46:48	01F0155N	2019-08-30 08:56:06	01F0029
127946	5	2019-08-30 08:17:24	01F3083S	2019-08-30 08:36:45	01F3126
237567	5	2019-08-30 08:08:02	03F0006S	2019-08-30 08:08:02	03F0006
237549	5	2019-08-30 08:51:27	03F0136S	2019-08-30 08:51:27	03F0136
15755	5	2019-08-30 08:52:16	01F1906S	2019-08-30 08:52:16	01F1906
197679	5	2019-08-30 08:31:55	03F1128S	2019-08-30 09:20:58	01F1725
197682	5	2019-08-30 08:59:28	01F0017S	2019-08-30 09:02:25	01F0061
197688	5	2019-08-30 08:40:56	01F3185S	2019-08-30 09:15:32	01F3686
164104	5	2019-08-30 08:20:36	03F1944S	2019-08-30 08:25:09	01F1960