Reimagining Public Transport, Post-Covid

Presentation Deck for
ST1510:Programming for Data Science (PDAS)
ST1502:Data Visualisation (DAVI)

Wong Zhao Wu

Background Context

The COVID-19 pandemic in Singapore is caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), which spreads primarily through droplets generated when an infected person coughs, sneezes or speaks. In response to the growing number of new community cases, Singapore enacted the COVID-19 Control Order, "circuit breaker", which caused a huge impact towards current public transport system and sparks the discussion over the reimagination for the public transport model to cater the change in commute pattern and in preparation for future epidemic spread by airborne transmission.

Objectives

1. Visualise Impacts of COVID-19 toward Singapore Public Transport System (Train & LRT, Taxi and Flight).

2. Investigate Solutions to Reimagine Singapore Public Transport System.

List of Raw Datasets

1.1 Train and LRT

- https://www.sbstransit.com.sg/ridership

1.2 Taxi

- https://www.lta.gov.sg/content/ltagov/en/who_we_are/statistics_and_publications/statistics.html

1.3 Flight

- https://data.gov.sg/dataset/civil-aircraft-arrivals-departures-passengers-and-mail-changi-airport-monthly
- https://www.changiairport.com/corporate/our-expertise/air-hub/traffic-statistics.html

2.1 Rethinking Micromobility: Promote Use of Micromobility Devices

- https://data.gov.sg/dataset/cycling-path-network
- https://data.gov.sg/dataset/hdb-cycling-paths-under-construction

2.2 Change Commuter Patterns: Stagger Work Hour

- http://datamall2.mytransport.sg/ltaodataservice/EstTravelTimes (DataMall Dynamic API: Full Documentation)
- https://developers.onemap.sg/commonapi/search (OneMap API: Full Documentation)

List of Cleaned Datasets

1 PublicTransport.csv

Columns	Descriptions	Example	DataType
Month	Month and Year of Records	1/1/2016 ... 1/12/2020	datetime64
SBSTransit Ridership	Ridership of Trains and LRTs operated by SBSTransit	875831	int64
Total Passengers	Total Number of Air Passengers in Changi Airport	4860156	float64
Average daily number of taxi trips(One-Shift)	Daily Average Number of Trips for One-Shift Taxi	18.9	float64
Average daily number of taxi trips(Two-Shift)	Daily Average Number of Trips for Two-Shift Taxi	28.6	float64
Average engaged mileage per trip (km)(One-Shift)	Average Taxi Mileage per Trip for One-Shift Taxi	10.1	float64
Average engaged mileage per trip (km)(Two-Shift)	Average Taxi Mileage per Trip for Two-Shift Taxi	9.6	float64

2 EstimatedTimeArrival.csv

Columns	Descriptions	Example	DataType
hour	Time of query of EST in 24h format	0,1,2...23	int64
Name	Name of Highway	AYE, BKE ... TPE	object
EndPoint	Name of Road EndPoint	TELOK BLANGAH RD	object
EstTime	Estimated Time Arrival towards EndPoint in Minute	2	float64
long	Longitude of EndPoint	103.810166483177	float64
lat	Latitude of EndPoint	1.27076318192364	float64

1.0 Visualise Impacts of COVID-19 toward Singapore Public Transport System (Train & LRT, Taxi and Flight).

1.1

Train & LRT

https://www.sbstransit.com.sg/ridership

Collecting Data

Ridership of Trains Operated by SBSTransit
(e.g. North East Line(NEL), Downtown Line(DTL), Sengkang and Punggol Light Rail Transit(LRT)) is scraped from SBSTransit website using ***requests*** and ***BeautifulSoup*** which is then parse to pandas Dataframe.

```python
html_soup = BeautifulSoup(requests.get('https://www.sbstransit.com.sg/ridership').text, 'html.parser') #Query HTML page and parse into BeautifulSoup obj
text = ''
for table_row in html_soup.find('table').find_all('tr')[1:]: #Iter over each row in html table
    text += table_row.get_text().replace(',','').replace('\n',',') #Remove comma in value and use comma as delimiter
    text += '\r\n' #Break into new line
train_df = pd.read_csv(StringIO(text)) #Turn String into DataFrame
train_df
```

	Unnamed: 0	Month	2014	2015	2016	2017	2018	2019	2020	2021	Unnamed: 10
0	NaN	January	649460	686426	875831	934363	1142026	1250833	1195131	844322	NaN
1	NaN	February	627959	691918	859465	948224	1141750	1186795	1016292		NaN
2	NaN	March	637010	688147	880966	955433	1144294	1217877	923319		NaN
3	NaN	April	637727	689305	879413	937046	1149121	1213490	288664		NaN
4	NaN	May	641864	696010	886145	925537	1146071	1201397	195515		NaN
5	NaN	June	654114	695964	882270	919224	1140105	1167470	421444		NaN
6	NaN	July	678218	738130	924305	970856	1222488	1273328	628983		NaN
7	NaN	August	687131	736760	931257	974022	1212155	1238783	685186		NaN
8	NaN	September	684510	713374	931944	949122	1210267	1236770	735858		NaN
9	NaN	October	676531	724470	924299	1010020	1221798	1241812	776078		NaN
10	NaN	November	668592	708426	903336	1102296	1182284	1213402	795843		NaN

Querying HTML Table and Extract the Information into Dataframe

```python
train_df.drop(columns = ['Unnamed: 0', 'Unnamed: 10', '2021', '2014', '2015'], inplace=True) #Drop Year 2014 to 2015 & 2021 to allow merging
train_df
```

	Month	2016	2017	2018	2019	2020
0	January	875831	934363	1142026	1250833	1195131
1	February	859465	948224	1141750	1186795	1016292
2	March	880966	955433	1144294	1217877	923319
3	April	879413	937046	1149121	1213490	288664
4	May	886145	925537	1146071	1201397	195515
5	June	882270	919224	1140105	1167470	421444
6	July	924305	970856	1222488	1273328	628983
7	August	931257	974022	1212155	1238783	685186
8	September	931944	949122	1210267	1236770	735858
9	October	924299	1010020	1221798	1241812	776078
10	November	903336	1102296	1182284	1213402	795843
11	December	908271	1079944	1152919	1170724	850271

Dropping Unwanted Columns

Transforming Data

Unpivoting Table and converting index column to `np.datetime64` format

```
train_df = train_df.melt(id_vars = 'Month', var_name='year', value_name='SBSTransit Ridership') #Unpivoting the dataframe
train_df.index = train_df[['Month', 'year']].agg(' '.join, axis = 1)#Combining Month and Year column and set it to index
train_df.index = pd.to_datetime(train_df.index, format="%B %Y")#Convert index into datetime format
train_df = train_df[['SBSTransit Ridership']]#Filter the dataframe so only SBSTransit Ridership column is left
train_df.head()
```

	SBSTransit Ridership
2016-01-01	875831
2016-02-01	859465
2016-03-01	880966
2016-04-01	879413
2016-05-01	886145

DataFrame Inspection

```
{}
```

DataFrame Inspection

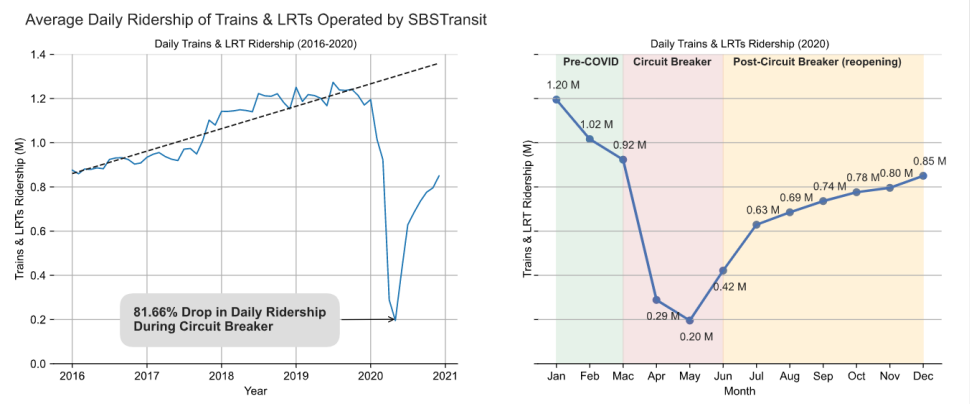
Observations:

```
All Columns is having correct datatypes of int64

Null Values is not observed in All Columns
```

```
train_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 60 entries, 2016-01-01 to 2020-12-01
Data columns (total 1 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   SBSTransit Ridership  60 non-null     int64
dtypes: int64(1)
memory usage: 960.0 bytes
```

Insights : Train & LRT



Average Lost in Daily Revenue for MRT and LRT operated by SBS Transit during Peak Circuit Breaker Period

From public-transport-utilisation-average-trip-distance, the average trip distance of Trains is around 10.5km which falls between range of 10.3km - 11.2km.

From Fares for MRT and LRT, adults fare at other timings with travel distance of 10.3km - 11.2km is 152 cents

$$S\$ \text{ Avg Lost in Daily Revenue} = (\text{Ridership}_{\text{Before COVID-19}} - \text{Ridership}_{\text{During Circuit Breaker}}) * 152 \text{ cents}$$

```
avg_adult_fare = 152/100 #152 Cent For Adult Fare of between range of 10.3km - 11.2km
ridership = pub_trans_df[pub_trans_df.Month < '2020-03-01']['SBSTransit Ridership']
print("Average Lost in Daily Revenue for MRT and LRT operated by SBS Transit during Peak Circuit Breaker Period:
\nS$ {:.2f} Daily".format(
    (pub_trans_df[pub_trans_df.Month < '2020-03-01']['SBSTransit Ridership'].mean()
    -pub_trans_df[pub_trans_df.Month >= '2020-03-01']['SBSTransit Ridership'].min()
    )*avg_adult_fare)
)
```

```
Average Lost in Daily Revenue for MRT and LRT operated by SBS Transit during Peak Circuit Breaker Period:
S$ 1325659.52 Daily
```

Observations

- 81.66% Drop in Daily Ridership of Train & LRT Operated by SBSTransit from around 1.06(M) Before COVID-19 to minimum of 0.20(M) During Peak COVID-19 Period.
- Average Lost in Daily Revenue during Peak COVID-19 period is around S$ 1325659.52. (By using Average Distance travelled at Normal Timing of Adult Prices)
- The same trend is observed from Busses and Other Trains Line, whereby daily ridership is reported to dropped 71% and 75% respectively. CNA Article

Key Insights

- Trains and LRT is heavily impacted by Covid-19 as it is practically impossible to implement safe distancing and indoor environment increases the risk of cross infection in public transport.
- Commuter can only leave house for very specific reason and businesses and schools are closed and move their activities online which reduces number of daily commuters.
  Circuit Breaker Rules
- Extra expenses for trains and LRT provider to increase the frequency of sanitization for trains and stations and to deploy more transport ambassador to ensure safe distancing. LTA NewsRoom

1.2

Taxi

https://www.lta.gov.sg/content/ltagov/en/who_we_are/statistics_and_publications/statistics.html

Collecting Data

Infos of Taxis Fleet in Singapore is obtained from LTA Statistics Taxi Columns.

The tables is extracted from **.pdf** format into **.csv** format with *tabula-py* converter and *manual* conversion.

```python
taxi_df = pd.read_csv('raw_data/Impact_of_COVID/taxi/taxi_info_2020.csv', index_col=0) #Reading First Dataset
for year in range(2019,2015,-1): #Loop Through Years from 2019 to 2016
    taxi_df_new = pd.read_csv(f'raw_data/Impact_of_COVID/taxi/taxi_info_{year}.csv', index_col=0) #Read Dataset of Next year
    taxi_df = pd.concat([taxi_df, taxi_df_new], axis = 1) #Append the records of dataset into next rows by column
taxi_df = taxi_df.T #Transpose the Dataset
taxi_df.head()
```

	Average daily number of taxi trips(One-Shift)	Average engaged mileage per trip (km)(One-Shift)	Average daily number of taxi trips(Two-Shift)	Average engaged mileage per trip (km)(Two-Shift)	Comfort	CityCab	Trans-Cab	SMRT	Premier	Prime	HDTT	Individual Yellow-Top Total	No. of TDVL issued	Total No. of valid TDVL holders
Jan-20	16.2	10.5	24.3	10.4	8106.0	2695.0	2821.0	2462.0	1576.0	672.0	129.0	67.0	226.0	101387.0
Feb-20	14.4	10.2	22.3	10.0	8121.0	2702.0	2711.0	2462.0	1560.0	675.0	129.0	65.0	90.0	101409.0
Mar-20	13.3	9.7	20.1	9.8	7988.0	2651.0	2498.0	2457.0	1524.0	665.0	129.0	65.0	181.0	101068.0
Apr-20	8.0	9.3	11.3	9.6	7685.0	2597.0	2373.0	2454.0	1446.0	665.0	129.0	65.0	15.0	100517.0
May-20	7.5	9.6	10.4	9.8	7588.0	2571.0	2328.0	1698.0	1334.0	665.0	129.0	65.0	202.0	99280.0

DataFrame Inspection

```
***Taxi Dataset Inspection***

<class 'pandas.core.frame.DataFrame'>
Index: 59 entries, Jan-20 to Dec-16
Data columns (total 14 columns):
 #   Column                                                Non-Null Count  Dtype
---  ------                                                --------------  -----
 0   Average daily number of taxi trips(One-Shift)         59 non-null     float64
 1   Average engaged mileage per trip (km)(One-Shift)      59 non-null     float64
 2   Average daily number of taxi trips(Two-Shift)         59 non-null     float64
 3   Average engaged mileage per trip (km)(Two-Shift)      59 non-null     float64
 4   Comfort                                               59 non-null     float64
 5   CityCab                                               59 non-null     float64
 6   Trans-Cab                                             59 non-null     float64
 7   SMRT                                                  59 non-null     float64
 8   Premier                                               59 non-null     float64
 9   Prime                                                 59 non-null     float64
 10  HDTT                                                  28 non-null     float64
 11  Individual Yellow-Top Total                           59 non-null     float64
 12  No. of TDVL issued                                    58 non-null     float64
 13  Total No. of valid TDVL holders                       59 non-null     float64
dtypes: float64(14)
memory usage: 6.9+ KB
```

- All Columns is having correct datatypes of float64
- Null Values is observed for HDTT & No. of TDVL issued columns.
- No Extreme Outliers is observed by comparing min and max values to 25% and 75% respectively. However further visualization is needed further investigation.

	Average daily number of taxi trips(One-Shift)	Average engaged mileage per trip (km)(One-Shift)	Average daily number of taxi trips(Two-Shift)	Average engaged mileage per trip (km)(Two-Shift)	Comfort	CityCab	Trans-Cab	SMRT	Premier	Prime	HDTT	Individual Yellow-Top Total	No. of TDVL issued	Total No. of valid TDVL holders
count	59.000000	59.000000	59.000000	59.000000	59.000000	59.000000	59.000000	59.000000	59.000000	59.000000	28.000000	59.000000	58.000000	59.000000
mean	16.647458	10.205085	25.030508	9.962712	9950.983051	3392.491525	3594.389831	2708.016949	1808.457627	696.677966	119.678571	89.084746	325.620690	99159.559322
std	2.303547	0.352050	3.516035	0.338301	1879.556032	657.611230	823.968300	613.591532	237.682730	41.291412	14.178893	27.764582	236.854521	1778.879154
min	7.500000	9.300000	10.400000	9.300000	7151.000000	2400.000000	2314.000000	1697.000000	1291.000000	622.000000	88.000000	57.000000	15.000000	95507.000000
25%	16.400000	10.000000	25.000000	9.750000	8318.000000	2817.000000	2961.500000	2298.000000	1673.000000	665.500000	108.250000	69.000000	169.750000	97910.500000
50%	17.200000	10.200000	25.600000	10.000000	9379.000000	3244.000000	3560.000000	2462.000000	1907.000000	687.000000	129.000000	81.000000	308.500000	99587.000000
75%	18.000000	10.450000	26.750000	10.200000	11880.000000	4075.000000	4549.500000	3386.000000	1985.000000	716.500000	129.000000	103.000000	397.250000	100475.500000
max	19.100000	10.800000	28.800000	10.500000	12775.000000	4372.000000	4847.000000	3577.000000	2062.000000	800.000000	129.000000	154.000000	1367.000000	101659.000000

Cleaning Null Values

Fill 0 for missing value of HDTT & No. of TDVL issued
(HDTT has been granted a taxi service operator licence starting from 1 Aug 2018)

```
print(f"***Before Imputing Null Values***\n{taxi_df.isnull().sum()}")
taxi_df.fillna(0, inplace = True)
print(f"\n***After Imputing Null Values***\n{taxi_df.isnull().sum()}")
```

```
***Before Imputing Null Values***
Average daily number of taxi trips(One-Shift)        0
Average engaged mileage per trip (km)(One-Shift)     0
Average daily number of taxi trips(Two-Shift)        0
Average engaged mileage per trip (km)(Two-Shift)     0
Comfort                                              0
CityCab                                              0
Trans-Cab                                            0
SMRT                                                 0
Premier                                              0
Prime                                                0
HDTT                                                31
Individual Yellow-Top Total                          0
No. of TDVL issued                                   1
Total No. of valid TDVL holders                      0
dtype: int64
```

```
***After Imputing Null Values***
Average daily number of taxi trips(One-Shift)        0
Average engaged mileage per trip (km)(One-Shift)     0
Average daily number of taxi trips(Two-Shift)        0
Average engaged mileage per trip (km)(Two-Shift)     0
Comfort                                              0
CityCab                                              0
Trans-Cab                                            0
SMRT                                                 0
Premier                                              0
Prime                                                0
HDTT                                                 0
Individual Yellow-Top Total                          0
No. of TDVL issued                                   0
Total No. of valid TDVL holders                      0
dtype: int64
```

Transforming Data

Converting Index to np.datetime64 data type

```
taxi_df.index = pd.to_datetime(taxi_df.index.str.replace('-',' '), format="%b %y") #Convert Index into Datetime format
taxi_df.index.dtype
```

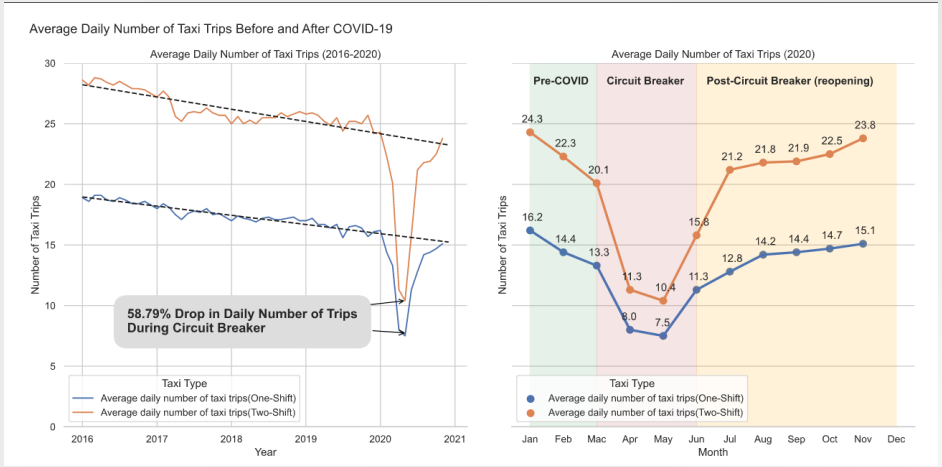
```
dtype('<M8[ns]')
```

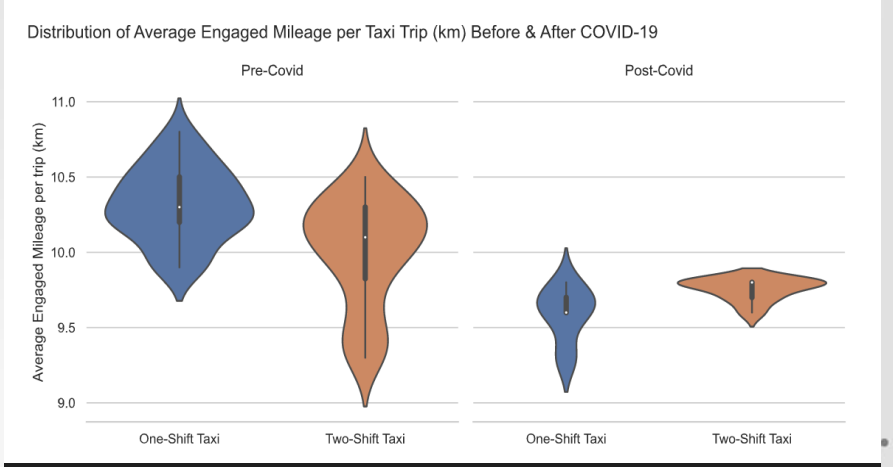
Dropping Unwanted Columns

```
taxi_df = taxi_df[['Average daily number of taxi trips(One-Shift)',
        'Average engaged mileage per trip (km)(One-Shift)',
        'Average daily number of taxi trips(Two-Shift)',
        'Average engaged mileage per trip (km)(Two-Shift)']]
taxi_df.head()
```

	Average daily number of taxi trips(One-Shift)	Average engaged mileage per trip (km) (One-Shift)	Average daily number of taxi trips(Two-Shift)	Average engaged mileage per trip (km) (Two-Shift)
2020-01-01	16.2	10.5	24.3	10.4
2020-02-01	14.4	10.2	22.3	10.0
2020-03-01	13.3	9.7	20.1	9.8
2020-04-01	8.0	9.3	11.3	9.6
2020-05-01	7.5	9.6	10.4	9.8

Insights : Taxi





Observations

- Average Number of Daily Trips for Taxi driver also face the same dropped of 58.79% in peak covid period from 16.2, 24.3 trips in Jan to 7.5, 10.4 trips in May.
- Average Mileage shows significant difference as the violin plot does not overlap and p-value of Two-sample independent t-test is $< \alpha=0.05$.
- Average Lost in Monthly Earning for Single Taxis during Peak COVID-19 Period for One-Shift Taxi and Two-Shift Taxi is S$ 1668.15 and S$ 2258.78.

Key Insights

- Taxi Industry is heavily impacted by COVID-19 as the number of daily trips and average engaged mileage shows a decline likely due to drop in number of commuters.
- Passengers are also skeptical of riding taxi they might assume the driver may contact many passengers in a day and taxis is not sanitized frequently which is prone to cross infection of COVID-19.

Two Sample Independent t-test for Average Mileage Before and After COVID-19

	P_Value	Test_Statistic	Degree_of_Freedom
Average Mileage (One-shift)	8.987793e-12	8.533827	57.0
Average Mileage (Two-shift)	4.498799e-02	2.049866	57.0

Average Lost in Monthly Earning for Single Taxis during Peak COVID-19 Period

Average Price rate of Normal Taxi is calculated based on following pricing formula

$$S\$\ \text{Avg Price} = 4(\text{Flag-Down Fares}) + \frac{\text{Avg Mileage(km)}}{0.4\text{km}} * 0.22(\text{Fares every 400m})$$

Average Lost in Monthly Earning for Single Taxis during Peak COVID-19 Period:
One-Shift Taxi: S$ 1668.15
Two-Shift Taxi: S$ 2258.78

1.3

Flight

https://data.gov.sg/dataset/civil-aircraft-arrivals-departures-passengers-and-mail-changi-airport-monthly

https://www.changiairport.com/corporate/our-expertise/air-hub/traffic-statistics.html

Collecting Data

Infos of Plane and Flights is obtained from Civil Aircraft Arrivals, Departures, Passengers And Mail, Changi Airport, Monthly at Data.gov.sg

Additional Data Number of Passengers in 2020 is obtained from Changi Airport Website

```
flight_df = pd.read_csv('raw_data/Impact_of_COVID/flight/
civil-aircraft-arrivals-departures-and-passengers-changi-airport-monthly.csv')
flight_df.head()
```

	month	level_1	value
0	1980-01	Total Aircraft Arrivals And Departures	6501
1	1980-01	Total Passengers	566879
2	1980-02	Total Aircraft Arrivals And Departures	6112
3	1980-02	Total Passengers	552263
4	1980-03	Total Aircraft Arrivals And Departures	6391

Use pd.pivot unstack flight_df into Columns by Months format

```
flight_df = pd.pivot(flight_df, columns='level_1', values='value', index='month')#Unstacking dataset
# Removing Name of Index(month) and Columns(level_1) after pivoting
flight_df.index.name=None
flight_df.columns.name=None
flight_df.head()
```

	Total Aircraft Arrivals And Departures	Total Passengers
1980-01	6501.0	566879.0
1980-02	6112.0	552263.0
1980-03	6391.0	597644.0
1980-04	6247.0	561218.0
1980-05	6301.0	587003.0

DataFrame Inspection

```
flight_df.info()
flight_df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 492 entries, 1980-01 to 2020-12
Data columns (total 2 columns):
 #   Column                                  Non-Null Count  Dtype
---  ------                                  --------------  -----
 0   Total Aircraft Arrivals And Departures  487 non-null    float64
 1   Total Passengers                        492 non-null    float64
dtypes: float64(2)
memory usage: 11.5+ KB
```

	Total Aircraft Arrivals And Departures	Total Passengers
count	487.000000	4.920000e+02
mean	15778.778234	2.446857e+06
std	8563.536552	1.506263e+06
min	3865.000000	2.450400e+04
25%	7463.500000	1.166619e+06
50%	14161.000000	2.199532e+06
75%	21144.500000	3.385848e+06
max	33435.000000	6.414495e+06

- All Columns is having correct datatypes of float64
- Null Values is observed in Total Aircraft Arrivals And Departures Columns
- No Extreme Outliers is observed by comparing min and max values to 25% and 75% respectively. However further visualization is needed further investigation.

Dropping Total Aircraft Arrivals and Departures Columns

```
flight_df.drop(columns='Total Aircraft Arrivals And Departures', inplace=True)
flight_df.columns
```

```
Index(['Total Passengers'], dtype='object')
```

Transforming Data

Converting Index to np.datetime64 data type

```
flight_df.index = pd.to_datetime(flight_df.index, format="%Y-%m")
flight_df.index.dtype
```

```
dtype('<M8[ns]')
```

Filter Rows to obtain records starting from year 2016 onwards

```
flight_df = flight_df.loc[flight_df.index.year>=2016]
flight_df.head()
```

	Total Passengers
2016-01-01	4860156.0
2016-02-01	4602026.0
2016-03-01	4902767.0
2016-04-01	4793662.0
2016-05-01	4781918.0

Merging Final Dataset

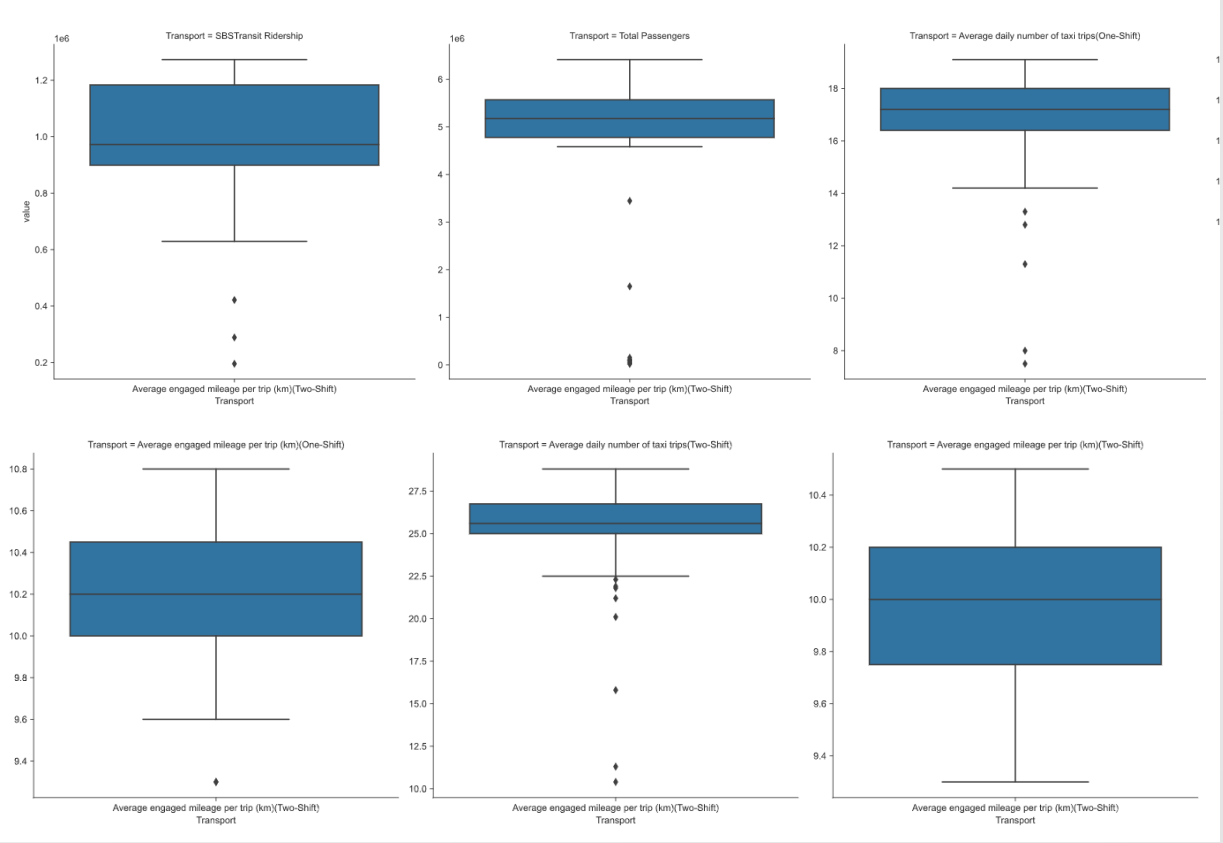
Merging Dataset Based on Index

Null Values is observed After Merging due to missing records, the values is not imputed for accurate visualisation

```
pub_trans_df = pd.merge(train_df, flight_df, left_index=True, right_index=True, how='outer')
pub_trans_df = pub_trans_df.merge(taxi_df, left_index=True, right_index=True, how='outer')
pub_trans_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 60 entries, 2016-01-01 to 2020-12-01
Freq: MS
Data columns (total 6 columns):
 #   Column                                          Non-Null Count  Dtype
---  ------                                          --------------  -----
 0   SBSTransit Ridership                            60 non-null     int64
 1   Total Passengers                                60 non-null     float64
 2   Average daily number of taxi trips(One-Shift)   59 non-null     float64
 3   Average engaged mileage per trip (km)(One-Shift) 59 non-null    float64
 4   Average daily number of taxi trips(Two-Shift)   59 non-null     float64
 5   Average engaged mileage per trip (km)(Two-Shift) 59 non-null    float64
dtypes: float64(5), int64(1)
memory usage: 3.3 KB
```

Checking Outliers



As most outliers are at the lower bound which is likely due to impact of COVID-19, imputing it might lose valuable information for data visualisation.

Hence, no action is taken to impute the outliers.

```
Exporting pub_trans_df to PublicTransport.csv

pub_trans_df.to_csv('cleaned_data/PublicTransport.csv', index_label='Month')
```

Insights : Flight

Air Passenger Movement in Changi Airport

Month	2016	2017	2018	2019	2020
Jan	4.9 M	5.3 M	5.3 M	5.7 M	6 M
Feb	4.6 M	4.7 M	4.9 M	5.1 M	3.4 M
Mac	4.9 M	5.1 M	5.6 M	5.6 M	1.6 M
Apr	4.8 M	5.2 M	5.4 M	5.6 M	0.025 M
May	4.8 M	5 M	5.3 M	5.4 M	0.025 M
Jun	4.8 M	5.2 M	5.6 M	5.8 M	0.048 M
Jul	5.2 M	5.4 M	5.7 M	5.9 M	0.086 M
Aug	4.9 M	5.3 M	5.7 M	5.9 M	0.084 M
Sep	4.6 M	4.9 M	5.2 M	5.5 M	0.09 M
Oct	4.8 M	5.2 M	5.4 M	5.6 M	0.098 M
Nov	4.8 M	5.2 M	5.4 M	5.7 M	0.11 M
Dec	5.7 M	5.9 M	6.1 M	6.4 M	0.15 M

Year

Observations

- Flight industry is impacted the most as Number of air passenger dropped from around 99.5% from around 5.3 M Before COVID-19 to merely 24.5 k in Peak COVID-19 Period
- Despite the lifting of Singapore Circuit Breaker Measures, number of air passengers still remained relatively low despite the common trend of spike in air travel in December from past few years.

Key Insights

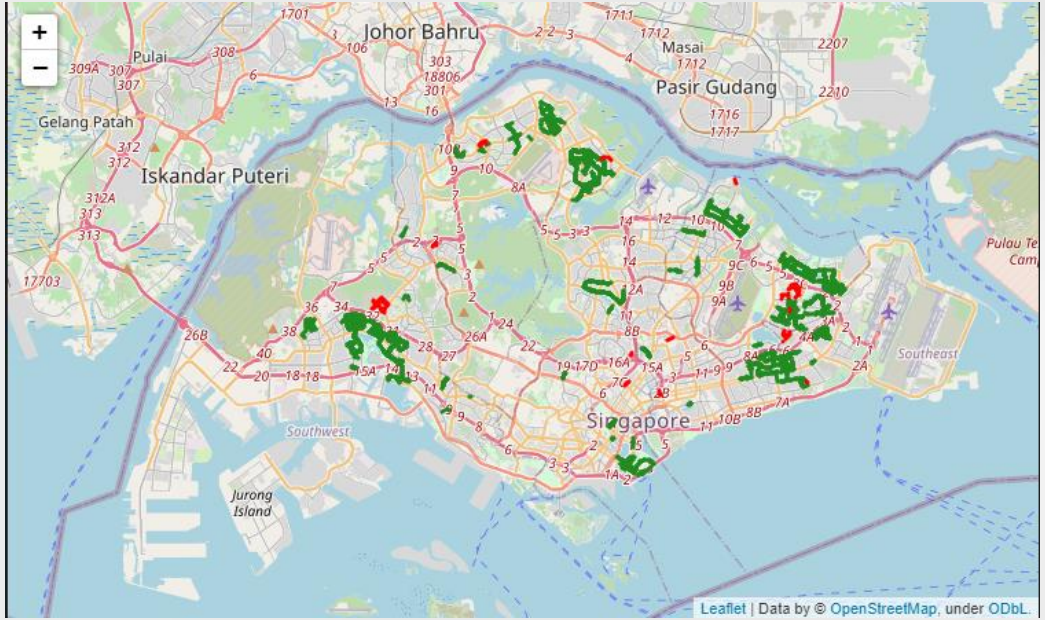
- Enclosed and Unventilated environment in a long flight is highly vulnerable to disease spread by airborne transmission including COVID-19 which leads to the decision of international travel restriction that is often carried out by planes.
- Although the situation of COVID-19 is stabalized in Singapore, the global condition is still highly unstable. Unless the travel restriction is lifted, the number of passengers will not return to Post-Covid period.

2.1

Rethinking Micromobility: Promote Use of Micromobility devices

https://data.gov.sg/dataset/cycling-path-network
https://data.gov.sg/dataset/hdb-cycling-paths-under-construction

Insights : Rethinking Micromobility: Promote Use of Micromobility devices



Insights

- Cycling paths in Singapore is very limited with only a few town have dedicated cycle paths which makes cycling unfriendly as cyclist have to either ride on road shoulders to avoid cars, or weave around pedestrians on pedestrian lanes.

- Lack of Cycling path connecting towns disregarding park connectors which is usually far from town centre that makes cycling inconvenience.

Recommendations

- Campaign to promote use of micromobility devices(*bicycles, electric bicycles, and motorised & non-motorised personal mobility devices (PMDs)*) to replace cross-towns and short distance travelling.

- Make roads cyclist friendly by dedicating left road lane in town to cyclist and build more cycling lane within and between towns.

Recommendations: Cycling Campaign

Campaign to promote use of micromobility devices(*bicycles, electric bicycles, and motorised & non-motorised personal mobility devices (PMDs)*) to replace cross-towns and short distance travelling.

SINGAPORE | TRANSPORT

E-scooters to be banned from Singapore footpaths from Nov 5

The Straits Times

Type of device	Footpaths (speed limit of 10kmh)	g/ shared paths (speed limit of 25kmh)	Roads
Personal mobility aid* e.g. motorised wheelchairs, mobility scooters	✔	✔	✖
Conventional bicycle			✔
E-scooters	✖		✖
LTA-approved power-assisted bicycle			✔

National Step Challenge : thesmartlocal

2.2

Change Commuter Patterns: Stagger Work Hour

http://datamall2.mytransport.sg/ltaodataservice/EstTravelTimes
(DataMall Dynamic API: Full Documentation)
https://developers.onemap.sg/commonapi/search
(OneMap API: Full Documentation)

Collecting Data

jupyter LTA DataMall EstTravelTime API Last Checkpoint: Last Thursday at 5:44 PM (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Run Code nbdiff

In [15]:
```
import requests
import pandas as pd
from datetime import datetime
import time
filename= "./EstTravelTime.csv"
```

Query LTA DataMall EstTravelTimes API

In [20]:
```
def query():
    try:
        response = requests.get(
            "http://datamall2.mytransport.sg/ltaodataservice/EstTravelTimes", # Query EstTravelTime from LTA DataMall
            headers = {"AccountKey":"g5ZPII+pQK6Yn95VAVQfaQ=="} # Unique AccountKey From DataMall
        )
        df_new = pd.DataFrame(response.json()['value']) #Convert Queried JSON to Dataframe
        df_new['time'] = datetime.now().strftime("%m/%d/%Y, %H:%M:%S") #Add Time Column(US Timezone as Using AWS SageMak
        df_prev = pd.read_csv(filename, index_col=0) #Import Previous Updated DataFrame
        df = pd.concat([df_prev, df_new], ignore_index=True) #Append New Records below Previous Updated DataFrame
        df.to_csv(filename) #Output Updated DataFrame and Replace the File
    except:
        print("Error")
```

In [ ]:
```
while True:
    query() #Query LTA DataMall API
    time.sleep(30*60) #Sleep for 30 Minutes to query API every 30 Minutes Interval
```

The Notebook Above is Run on AWS SageMaker to Query LTA DataMall EstTravelTimes API for 24 Hour with 30 Minutes Interval on a Normal Weekday and Output Is Saved as raw_data/EstTravelTimes.csv

```
eta_df = pd.read_csv('raw_data/EstTravelTime.csv', index_col=0)
eta_df
```

	Name	Direction	FarEndPoint	StartPoint	EndPoint	EstTime	time
0	AYE	1	TUAS CHECKPOINT	AYE/MCE INTERCHANGE	TELOK BLANGAH RD	2	01/28/2021, 10:00:00
1	AYE	1	TUAS CHECKPOINT	TELOK BLANGAH RD	LOWER DELTA RD	1	01/28/2021, 10:00:00
2	AYE	1	TUAS CHECKPOINT	LOWER DELTA RD	NORMANTON PARK	4	01/28/2021, 10:00:00
3	AYE	1	TUAS CHECKPOINT	NORMANTON PARK	NORTH BUONA VISTA RD	2	01/28/2021, 10:00:00
4	AYE	1	TUAS CHECKPOINT	NORTH BUONA VISTA RD	CLEMENTI RD	2	01/28/2021, 10:00:00
...	...	...	...	...	...	...	...
8587	TPE	2	PIE	PASIR RIS DRIVE 12	ELIAS ROAD	1	01/29/2021, 09:30:50
8588	TPE	2	PIE	ELIAS ROAD	PASIR RIS DRIVE 8	1	01/29/2021, 09:30:50
8589	TPE	2	PIE	PASIR RIS DRIVE 8	TAMPINES AVE 7	2	01/29/2021, 09:30:50
8590	TPE	2	PIE	TAMPINES AVE 7	LOYANG AVE	1	01/29/2021, 09:30:50
8591	TPE	2	PIE	LOYANG AVE	PIE/TPE INTERCHANGE	1	01/29/2021, 09:30:50

Transforming Data

Convert DateTime from UTC to Singapore Timezone (UTC+08:00)

```
eta_df['time'] = pd.to_datetime(eta_df.time).dt.tz_localize('utc').dt.tz_convert(pytz.timezone('Asia/Singapore'))
 # Covert DateTime from UTC to Singapore Timezone
eta_df['hour'] = eta_df.time.dt.hour # Creating Hour column
```

Query OneMap API To Obtain Longitude and Latitude of Road

OneMap API is unable to get the Longitude and Latitude for all roads (Interchange between highways), np.nan is use to fill the gap of null values before cleaning

```
unique_road = eta_df.EndPoint.unique() # Get Unique Endpoint
unique_road_loc_arr = [] # Array that store the dicts of Longitude and Latitude of Endpoint(Road)

for road in unique_road:
    query_string='https://developers.onemap.sg/commonapi/search?searchVal='+str(road)+'&returnGeom=Y&getAddrDetails=Y&pageNum=1'
    resp = requests.get(query_string) # Query OpenMap API to obtain Longitude and Latitude of Endpoint
    try:
        data=resp.json()
        unique_road_loc_arr.append( # Append Longitude and Latitude Dict to Arr
            dict(
                EndPoint=road,
                long=data['results'][0]['LONGITUDE'],
                lat=data['results'][0]['LATITUDE']
                )
            )
    except:
        print ("{} Error".format(road))
        unique_road_loc_arr.append( # Append np.nan when the road is not found
            dict(
                EndPoint=road,
                long=np.nan,
                lat=np.nan
                )
            )
        pass
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   EndPoint  119 non-null    object
 1   long      95 non-null     object
 2   lat       95 non-null     object
dtypes: object(3)
memory usage: 2.9+ KB


***After Converting DataType of Long & Lat***

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 119 entries, 0 to 118
Data columns (total 3 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   EndPoint  119 non-null    object
 1   long      95 non-null     float64
 2   lat       95 non-null     float64
dtypes: float64(2), object(1)
memory usage: 2.9+ KB
```

Transforming Data

Merging eta_df(queried from LTA DataMall) with unique_road_loc_df(queried from OneMap)

```
eta_loc_df = pd.merge(eta_df, unique_road_loc_df, on='EndPoint', how='left') # Merge Dataframe on EndPoint
eta_loc_df = eta_loc_df.groupby( # Perform Grouping and Obtain the Mean of EstTime(From 2 Direction and multiple timeframe)
    ['hour','Name','EndPoint'], as_index = False, sort=False # Sort = False to preserve order of EndPoint(For Filling Null with ffill method)
    )[['EstTime','long','lat']].mean().sort_values(['hour','Name']) # Sort according to hour and Name
eta_loc_df.head()
```

	hour	Name	EndPoint	EstTime	long	lat
768	0	AYE	TELOK BLANGAH RD	2.0	103.810166	1.270763
769	0	AYE	LOWER DELTA RD	1.5	103.823633	1.279961
770	0	AYE	NORMANTON PARK	3.0	103.792795	1.287235
771	0	AYE	NORTH BUONA VISTA RD	1.5	103.790509	1.307025
772	0	AYE	CLEMENTI RD	1.0	103.778487	1.337007

Imputing Null Values for Longitude and Latitude with FrontFill Method

```
print(f"***Before Imputing Null Values***\n{eta_loc_df.isnull().sum()}")
eta_loc_df.fillna(method='ffill', inplace = True) # FrontFill Null Values for Longitude and Latitude
print(f"\n***After Imputing Null Values***\n{eta_loc_df.isnull().sum()}")
```

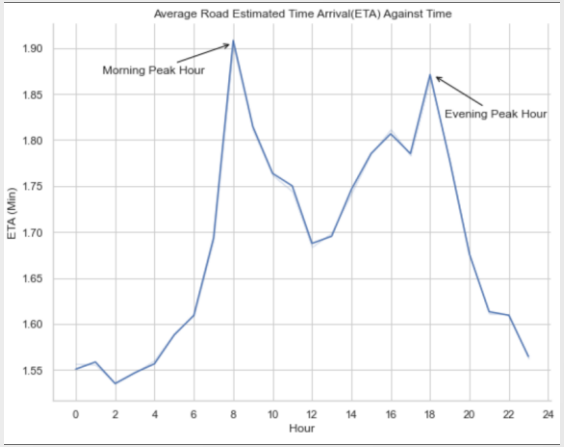
```
***Before Imputing Null Values***
hour          0
Name          0
EndPoint      0
EstTime       0
long        648
lat         648
dtype: int64

***After Imputing Null Values***
hour        0
Name        0
EndPoint    0
EstTime     0
long        0
lat         0
dtype: int64
```

Standardization for Estimated Time Arrival

Standardize Each Endpoint Across 24 Hours to capture the variability of ETA across Endpoints and better Visualise the Congestion across Time with Following Formula:

$$Est_{standardized} = \frac{Est - \mu_{Est}}{\sigma_{Est}}$$

Insights : Change Commuter Patterns: Stagger Work Hour

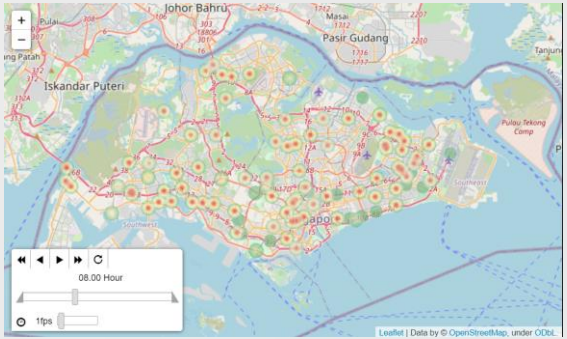


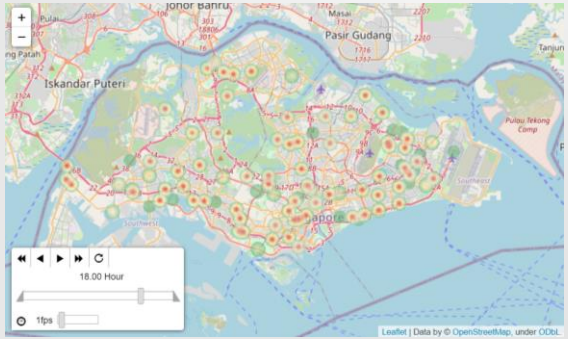
Insights

- From Average Estimated Time Arrival of major highways, we can observe spike in mean ETA in morning(8.00am) and evening(6.00pm) hour.
- From heat bubble map, trend of peak hour can be observe island wide at morning(8.00am) and evening(6.00pm) hour which reflects to the standard working hours of company, results in highly condensed highways and presumably public transports(Trains, LRTs, Busses).

Recommendations

- Staggering Working hours in company and digitalising operation of company to flatten the curve of passenger density of public transport.
- Increase the frequency of trains & busses and expansion of train networks to lower the passenger volume of trains & busses.





Recommendations: Stagger Work Hour

Staggering Working hours in company and digitalising operation of company to flatten the curve of passenger density of public transport.

Increase the frequency of trains & busses and expansion of train networks to lower the passenger volume of trains & busses.

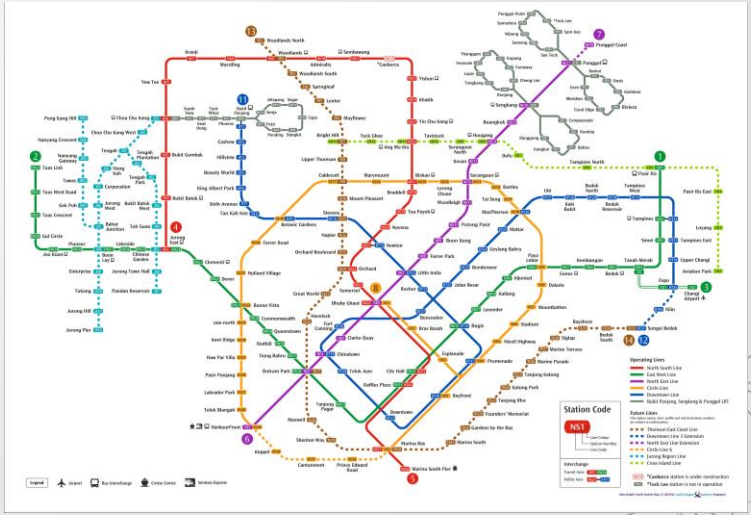
SINGAPORE

Coronavirus Singapore

Work from home to stay as default to lower office transmission risk

Tripartite partners issue update after review of safe management measures at workplace

The Straits Times



LTA: MRT lines in 2030

Conclusion

The short-term heavy impact towards public transport system clearly indicates its flaw towards epidemic caused by airborne transmission.

Even if COVID-19 is stabilized by current vaccination program in Singapore, article have shown that it will take at least 7 years before life can turn back to pre-Covid norm on a global scale and the risk of another highly infectious epidemic caused by airborne transmission cannot be neglected.

We should learn from this experience and reimagining current public transport system by -promoting micromobility and staggering work hours- such that impact towards public transport model can be mitigated.

THANKS

Does anyone have any questions?