



## School of Computing

**SRM IST, Kattankulathur – 603 203**

**Course Code: 18CSC206J**

**Course Name: Software Engineering and Project Management**



<b>Experiment No</b>	13
<b>Title of Experiment</b>	Provide the details of Architecture Design/Framework/Implementation
<b>Name of the candidate</b>	Aniket Mishra
<b>Team Members</b>	G. Avinashh, Prachi Pandit
<b>Team Name</b>	Find Your Air (F.Y.A)
<b>Register Number</b>	RA2011003010098
<b>Date of Experiment</b>	20.6.22

### Mark Split Up

<b>S. No</b>	<b>Description</b>	<b>Maximum Mark</b>	<b>Mark Obtained</b>
1	Exercise	5	
2	Viva	5	
<b>Total</b>		<b>10</b>	

**Staff Signature with date**

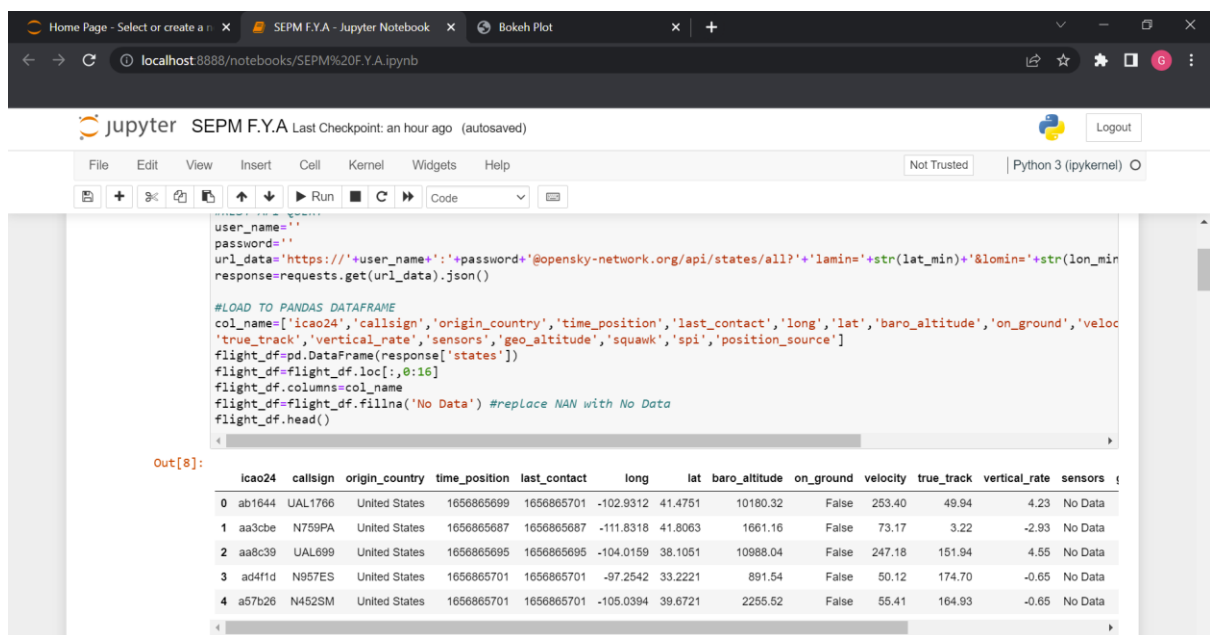
## Aim:

To provide the details of architectural design/framework/implementation

## Team Members:

S No	Register No	Name	Role
1	RA2011003010085	Prachi Pandit	Rep/Member
2	RA2011003010080	G. Avinashh	Member
3	RA2011003010098	Aniket Mishra	Member

## Screenshots of implementation:



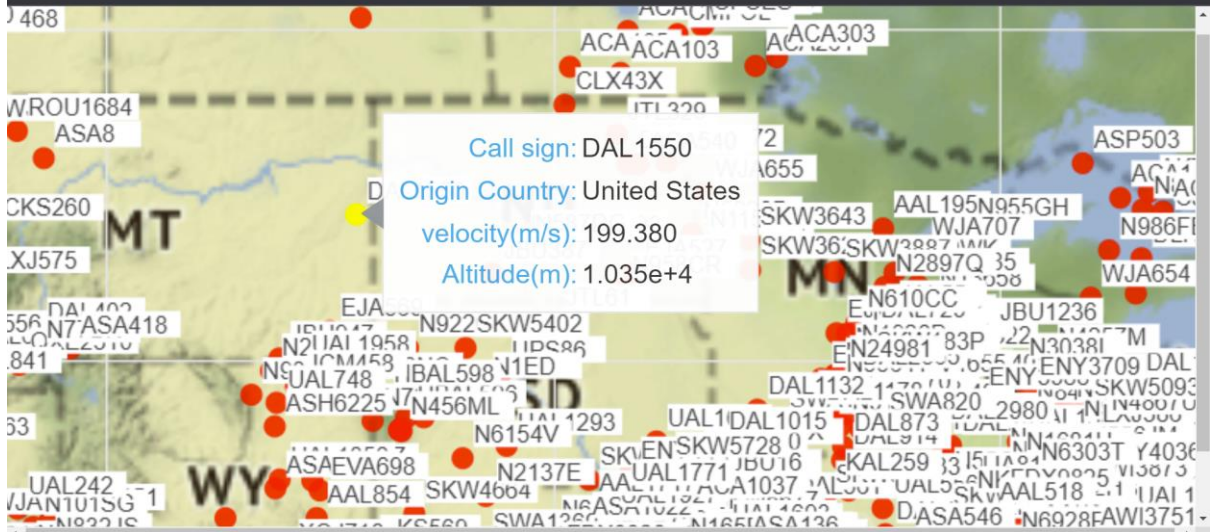
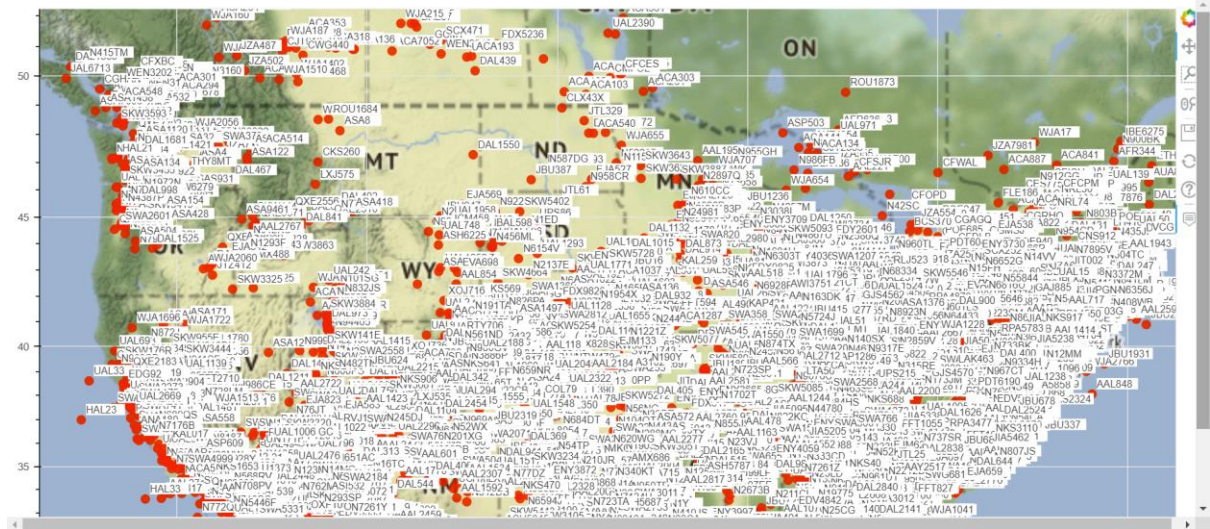
The screenshot displays a Jupyter Notebook interface with a dark theme. The browser address bar shows the URL `localhost:8888/notebooks/SEPM%20F.Y.A.ipynb`. The notebook title is "SEPM F.Y.A" and it indicates the last checkpoint was saved an hour ago. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations, running, and code execution. The code cell contains a Python script that uses the `requests` library to fetch data from an API and the `pandas` library to process it. The output of the code is displayed as a table with 12 columns and 5 rows of data.

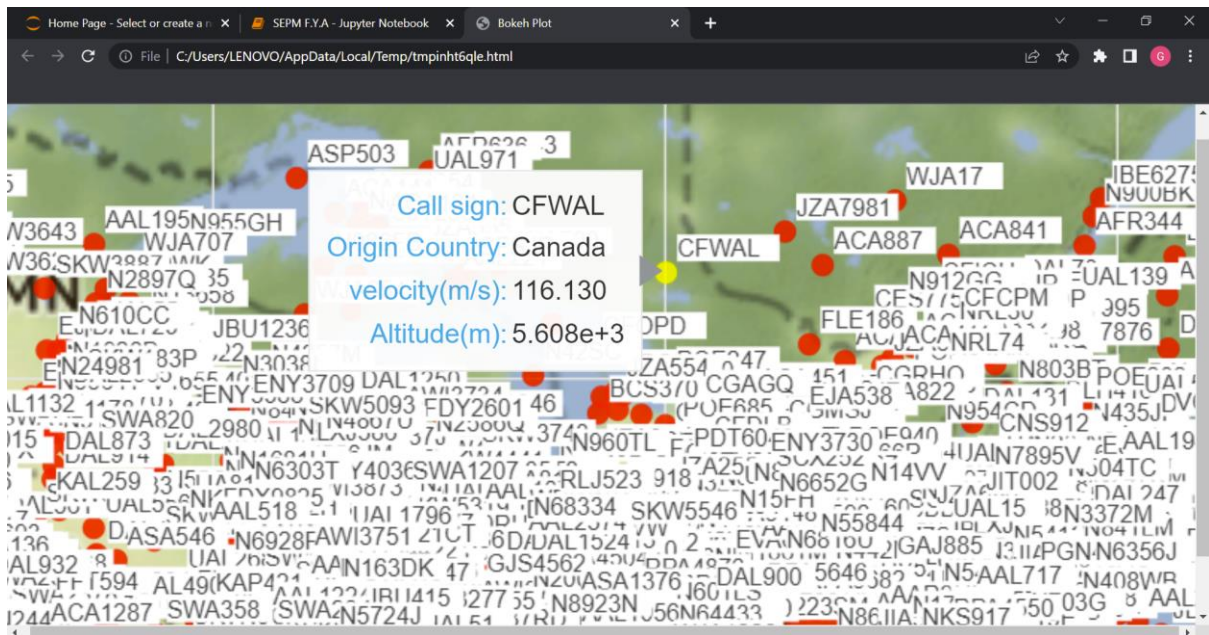
```
user_name=""
password=""
url_data='https://'+user_name+'-'+password+'@opensky-network.org/api/states/all?'+lamin='+str(lat_min)+'&lomin='+str(lon_min)
response=requests.get(url_data).json()

#LOAD TO PANDAS DATAFRAME
col_name=['icao24','callsign','origin_country','time_position','last_contact','long','lat','baro_altitude','on_ground','velocity','true_track','vertical_rate','sensors','geo_altitude','squawk','spi','position_source']
flight_df=pd.DataFrame(response['states'])
flight_df=flight_df.loc[:,0:16]
flight_df.columns=col_name
flight_df=flight_df.fillna('No Data') #replacE NAN with No Data
flight_df.head()
```

Out[8]:

	icao24	callsign	origin_country	time_position	last_contact	long	lat	baro_altitude	on_ground	velocity	true_track	vertical_rate	sensors
0	ab1644	UAL1766	United States	1656865699	1656865701	-102.9312	41.4751	10180.32	False	253.40	49.94	4.23	No Data
1	aa3cbe	N759PA	United States	1656865687	1656865687	-111.8318	41.8063	1661.16	False	73.17	3.22	-2.93	No Data
2	aa8c39	UAL699	United States	1656865695	1656865695	-104.0159	38.1051	10988.04	False	247.18	151.94	4.55	No Data
3	ad411d	N957ES	United States	1656865701	1656865701	-97.2542	33.2221	891.54	False	50.12	174.70	-0.65	No Data
4	a57b26	N452SM	United States	1656865701	1656865701	-105.0394	39.6721	2255.52	False	55.41	164.93	-0.65	No Data





## Code:

```

#IMPORTING LIBRARY
import requests
import json
import pandas as pd

#AREA EXTENT COORDINATE WGS4
lon_min,lat_min=-125.974,30.038
lon_max,lat_max=-68.748,52.214

#REST API QUERY
user_name=''
password=''
url_data='https://'+user_name+':'+password+'@opensky-network.org/api/states/all?'+lamin='+str(lat_min)+'&lomin='+str(lon_min)
response=requests.get(url_data).json()

#LOAD TO PANDAS DATAFRAME
col_name=['icao24','callsign','origin_country','time_position','last_contact','long','lat','baro_altitude','on_ground','veloc
'true_track','vertical_rate','sensors','geo_altitude','squawk','spi','position_source']
flight_df=pd.DataFrame(response['states'])
flight_df=flight_df.loc[:,0:16]
flight_df.columns=col_name
flight_df=flight_df.fillna('No Data') #replce NAN with No Data
flight_df.head()

```

```
Home Page - Select or create a notebook | SEPM F.Y.A - Jupyter Notebook | +
localhost8888/notebooks/SEPM%20F.Y.A.ipynb
jupyter SEPM F.Y.A Last Checkpoint: an hour ago (autosaved) Logout
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)
In [9]: #IMPORT PLOTTING LIBRARIES
from bokeh.plotting import figure, show
from bokeh.tile_providers import get_provider, STAMEN_TERRAIN
from bokeh.models import HoverTool, LabelSet, ColumnDataSource
import numpy as np

#FUNCTION TO CONVERT GCS WGS84 TO WEB MERCATOR
#POINT
def wgs84_web_mercator_point(lon,lat):
    k = 6378137
    x= lon * (k * np.pi/180.0)
    y= np.log(np.tan((90 + lat) * np.pi/360.0)) * k
    return x,y

#DATA FRAME
def wgs84_to_web_mercator(df, lon="long", lat="lat"):
    k = 6378137
    df["x"] = df[lon] * (k * np.pi/180.0)
    df["y"] = np.log(np.tan((90 + df[lat]) * np.pi/360.0)) * k
    return df

#COORDINATE CONVERSION
xy_min=wgs84_web_mercator_point(lon_min,lat_min)
xy_max=wgs84_web_mercator_point(lon_max,lat_max)
wgs84_to_web_mercator(flight_df)
flight_df['rot_angle']=flight_df['true_track']*1 #Rotation angle
```

```
Home Page - Select or create a notebook | SEPM F.Y.A - Jupyter Notebook | +
localhost8888/notebooks/SEPM%20F.Y.A.ipynb
jupyter SEPM F.Y.A Last Checkpoint: an hour ago (autosaved) Logout
File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)
wgs84_to_web_mercator(flight_df)
flight_df['rot_angle']=flight_df['true_track']*1 #Rotation angle
icon_url='https://.....' #Icon url
flight_df['url']=icon_url

#FIGURE SETTING
x_range,y_range=([xy_min[0],xy_max[0]], [xy_min[1],xy_max[1]])
p=figure(x_range=x_range,y_range=y_range,x_axis_type='mercator',y_axis_type='mercator',sizing_mode='scale_width',plot_height=

#PLOT BASEMAP AND AIRPLANE POINTS
flight_source=ColumnDataSource(flight_df)
tile_prov=get_provider(STAMEN_TERRAIN)
p.add_tile(tile_prov,level='image')
p.image_url(url='url', x='x', y='y',source=flight_source,anchor='center',angle_units='deg',angle='rot_angle',h_units='screen'
p.circle('x','y',source=flight_source,fill_color='red',hover_color='yellow',size=10,fill_alpha=0.8,line_width=0)

#HOVER INFORMATION AND LABEL
my_hover=HoverTool()
my_hover.tooltips=[('Call sign','@callsign'),('Origin Country','@origin_country'),('velocity(m/s)','@velocity'),('Altitude(m)
labels = LabelSet(x='x', y='y', text='callsign', level='glyph',
x_offset=5, y_offset=5, source=flight_source, render_mode='canvas',background_fill_color='white',text_font_size="
p.add_tools(my_hover)
p.add_layout(labels)

show(p)
```



The screenshot displays a Jupyter Notebook window in a web browser. The browser's address bar shows the URL `localhost:8888/notebooks/SEPM%20F.Y.A.ipynb`. The Jupyter interface includes a top bar with the notebook title "SEPM F.Y.A" and a "Last Checkpoint: an hour ago (autosaved)" message. Below this is a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. A toolbar with icons for saving, running, and other actions is also present. The main area shows a code cell with the following Python code:

```
In [10]: #IMPORT LIBRARY
import requests
import json
import pandas as pd
from bokeh.plotting import figure
from bokeh.models import HoverTool, LabelSet, ColumnDataSource
from bokeh.tile_providers import get_provider, STAMEN_TERRAIN
import numpy as np
from bokeh.server.server import Server
from bokeh.application import Application
from bokeh.application.handlers.function import FunctionHandler

#FUNCTION TO CONVERT GCS WGS84 TO WEB MERCATOR
#DATAFRAME
def wgs84_to_web_mercator(df, lon="long", lat="lat"):
    k = 6378137
    df["x"] = df[lon] * (k * np.pi/180.0)
    df["y"] = np.log(np.tan((90 + df[lat]) * np.pi/360.0)) * k
    return df

#POINT
def wgs84_web_mercator_point(lon,lat):
    k = 6378137
    x= lon * (k * np.pi/180.0)
    y= np.log(np.tan((90 + lat) * np.pi/360.0)) * k
    return x,y
```

Result:

Thus, the details of architectural design/framework/implementation along with the screenshots were provided.