**import** plotly.express **as** px

**import** pandas **as** pd

**import** plotly.graph\_objects **as** go

**import** streamlit **as** st

**from** plotly.subplots **import** make\_subplots

**import** datetime

**import** numpy **as** np

**from** ipywidgets **import** widgets

**import** time  *# to simulate a real time data, time loop*

**import** numpy **as** np  *# np mean, np random*

**import** pandas **as** pd  *# read csv, df manipulation*

**import** plotly.express **as** px  *# interactive charts*

**import** streamlit **as** st  *# 🎈 data web app development*

**import** numpy **as** np

**import** pandas **as** pd

**from** sklearn.preprocessing **import** RobustScaler

**from** scipy **import** stats

**from** sklearn.preprocessing **import** OneHotEncoder

**import** keras

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** tensorflow **as** tf

**import** numpy **as** np

**import** pandas **as** pd

**from** sklearn.preprocessing **import** RobustScaler

**from** scipy **import** stats

**from** sklearn.preprocessing **import** OneHotEncoder

**import** keras

**import** matplotlib.pyplot **as** plt

**import** seaborn **as** sns

**import** tensorflow **as** tf

st.set\_page\_config(

**page\_title=**"ESP Gas Lock Detection",

**page\_icon=**"https://upload.wikimedia.org/wikipedia/commons/thumb/d/d6/SLB\_Logo\_2022.svg/640px-SLB\_Logo\_2022.svg.png",

**layout=**"wide",

)

st.sidebar.header('User Input Features')

*# Collects user input features into dataframe*

uploaded\_file **=** st.sidebar.file\_uploader("Upload your input CSV file", **type=**["csv"])

st.sidebar.subheader("Run the Program")

result **=** st.sidebar.button("Click Here")

st.write("The Program Can Run With or Without a CSV Upload")

**if** uploaded\_file **is** **not** None **and** result:

    esp3 **=** pd.read\_csv(uploaded\_file)

    esp **=** esp3.copy()

    espj **=** esp3.copy()

    espc **=** esp3.copy()

    espd **=** esp3.copy()

**else**:

*#esp3 = pd.read\_csv("espjanuari.csv")*

    esp3 **=** pd.read\_csv("esphelp1.csv")

*#bb = pd.read\_csv("esppetang3.csv")*

    esp **=** esp3.copy()

    espj **=** esp3.copy()

    espc **=** esp3.copy()

    espd **=** esp3.copy()

**if** result:

*#Data Awal*

    esp2 **=** esp.copy() *#Penampil Waktu Box*

    esp4 **=** esp2.copy()

    esp.timestamp **=** pd.to\_datetime(esp.timestamp)

*#Data Pembagian*

    espn **=** esp2.iloc[0:1, :]

    espg **=** esp2.iloc[0:1, :]

**with** open('style.css') **as** f:

        st.markdown(f'<style>{f.read()}</style>', **unsafe\_allow\_html=**True)

*# read csv from a URL*

*# dashboard title*

    st.title("ESP Gas Lock Detection Dashboard")

*# creating a single-element container*

    placeholder **=** st.empty()

*#MODEL INITIALIZE*

    new\_model **=** tf.keras.models.load\_model('saved\_modelik1/my\_modelik1')

*#DATASET BARU*

**def** create\_dataset(**X**, **time\_steps=**1, **step=**1):

        Xs **=** []

**for** i **in** range(0, len(X), step):

*#for i in range(0, 1881 - 11, step):*

            v **=** X.iloc[i:(i **+** time\_steps)].values

*#v = X.iloc[0 : 0 + 11].values*

*#print(labels)*

*#labels = y.iloc[0 : 0 + 11]*

            Xs.append(v)

**return** np.array(Xs)

*# near real-time / live feed simulation*

    x **=** int(len(esp3)**/**11)

    xx **=** x

    a **=** 0; b **=** 11

    h **=** 1; m **=** 2

    d **=** 1; e **=** 2; f **=** 2

    q **=** 0

    espc **=** espc.iloc[a:b**\***x,:]

    zurface **=** []

    yurface **=** []

**for** i **in** range(0, xx):

*#esp3 = pd.read\_csv("espjanuari.csv")*

        esp3 **=** pd.read\_csv("esphelp1.csv")

        esp4 **=** esp3.iloc[0:b,:]

        espj **=** esp3.iloc[a:b,:]

        espjc **=** espj.copy()

        espj **=** pd.concat([espd, espj])

*#MODEL*

*#SCALER*

        scale\_columns **=** ["Motor Temperature(F)", "Intake Pressure(psi)", "Discharge Pressure(psi)",

                        "Average Amps(Amps)", "Intake Temperature(F)", "Drive Frequency(Hz)"]

        scaler **=** RobustScaler()

        scaler **=** scaler.fit(espj[scale\_columns])

        espj.loc[:, scale\_columns] **=** scaler.transform(

        espj[scale\_columns].to\_numpy()

        )

*#STEPS BARU*

        TIME\_STEPS **=** 11

        STEP **=** 11

        X **=** create\_dataset(

            espj[["Motor Temperature(F)", "Intake Pressure(psi)", "Discharge Pressure(psi)",

                        "Average Amps(Amps)", "Intake Temperature(F)", "Drive Frequency(Hz)"]],

            TIME\_STEPS,

            STEP

        )

*#Encrypt*

        enc **=** OneHotEncoder(**handle\_unknown=**'ignore', **sparse=**False)

        X **=** np.asarray(X).astype(np.float32)

*#tf.convert\_to\_tensor(X, dtype=tf.float32)*

        y\_pred **=** new\_model.predict(X)

        y\_pred **=** np.argmax(y\_pred, **axis=**1)

        surface **=** []

**for** ip **in** y\_pred[**-**1:]:

**if** ip **==** 0:

                surface.append("Gas Lock Detected")

**elif** ip **==** 1:

                surface.append("Normal Condition")

        zurface **=** []

**for** ix **in** surface[**-**1:]:

**for** j **in** range(0, 11):

                zurface.append(ix)

**for** iy **in** zurface:

            yurface.append(iy)

        esp4["State"] **=** yurface

*#dif["State"] = zurface*

        espjc["State"] **=** zurface

**if** zurface[**-**1] **==** "Gas Lock Detected":

            espg **=** pd.concat([espg, espjc])

            espg.timestamp **=** pd.to\_datetime(espg.timestamp)

**elif** zurface[**-**1] **==** "Normal Condition":

            espn **=** pd.concat([espn, espjc])

            espn.timestamp **=** pd.to\_datetime(espn.timestamp)

*#espt = pd.concat(espt, espjc)*

**for** seconds **in** range(11):

*#h += 1*

            m **+=** 1

**with** placeholder.container():

                kpi1, kpi2 **=** st.columns(2)

                kpi1.metric(

**label=**"Timestamp",

**value=**esp2["timestamp"][m]

                )

                kpi2.metric(

**label=**"State",

**value=**espjc["State"][q]

                )

*# create three columns*

                kpi1, kpi2, kpi3 **=** st.columns(3)

*# fill in those three columns with respective metrics or KPIs*

                kpi1.metric(

**label=**"Discharge Pressure (psi)",

**value=**int(esp["Discharge Pressure(psi)"][m]),

**delta=**esp["Discharge Pressure(psi)"][m] **-** esp["Discharge Pressure(psi)"][m**-**1],

                )

                kpi2.metric(

**label=**"Average Amps (Amp)",

**value=**int(esp["Average Amps(Amps)"][m]),

**delta=**esp["Average Amps(Amps)"][m] **-** esp["Average Amps(Amps)"][m**-**1],

                )

                kpi3.metric(

**label=**"Intake Temperature(F)",

**value=**int(esp["Intake Temperature(F)"][m]),

**delta=**esp["Intake Temperature(F)"][m] **-** esp["Intake Temperature(F)"][m**-**1],

                )

*# create three columns*

                kpi1, kpi2, kpi3 **=** st.columns(3)

*# fill in those three columns with respective metrics or KPIs*

                kpi1.metric(

**label=**"Drive Frequency(Hz)",

**value=**int(esp["Drive Frequency(Hz)"][m]),

**delta=**esp["Drive Frequency(Hz)"][m] **-** esp["Drive Frequency(Hz)"][m**-**1],

                )

                kpi2.metric(

**label=**"Motor Temperature(F)",

**value=**int(esp["Motor Temperature(F)"][m]),

**delta=**esp["Motor Temperature(F)"][m] **-** esp["Motor Temperature(F)"][m**-**1],

                )

                kpi3.metric(

**label=**"Intake Pressure(psi)",

**value=**int(esp["Intake Pressure(psi)"][m]),

**delta=**esp["Intake Pressure(psi)"][m] **-** esp["Intake Pressure(psi)"][m**-**1],

                )

*#GRAPH FULL*

                st.markdown("# ESP DATA")

                fig **=** go.Figure()

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Discharge Pressure(psi)"][h**-**1:m**-**1],

**name=**"Discharge Pressure(psi)", **mode=**"lines", **line\_color=**"#FF00EE"

                ))

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Average Amps(Amps)"][h**-**1:m**-**1],

**name=**"Average Amps(Amps)",

**yaxis=**"y2", **mode=**"lines", **line\_color=**"#AB5959"

                ))

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Intake Temperature(F)"][h**-**1:m**-**1],

**name=**"Intake Temperature(F)",

**yaxis=**"y3", **mode=**"lines", **line\_color=**"#060CBD"

                ))

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Drive Frequency(Hz)"][h**-**1:m**-**1],

**name=**"Drive Frequency(Hz)",

**yaxis=**"y4", **mode=**"lines", **line\_color=**"#96A35C"

                ))

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Motor Temperature(F)"][h**-**1:m**-**1],

**name=**"Motor Temperature(F)",

**yaxis=**"y5", **mode=**"lines", **line\_color=**"#006B33"

                ))

                fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h**-**1:m**-**1],

**y=**esp["Intake Pressure(psi)"][h**-**1:m**-**1],

**name=**"Intake Pressure(psi)",

**yaxis=**"y6", **mode=**"lines", **line\_color=**"#98FFFD"

                ))

*# Create axis objects*

                fig.update\_layout(

**xaxis=**dict(

**domain=**[0.075, 0.95]

                    ),

**autosize=**False,

**width=**3000,

**height=**2500,

**yaxis=**dict(

**titlefont=**dict(

**color=**"#FF00EE"

                        ),

**tickfont=**dict(

**color=**"#FF00EE"

                        )

                    ),

**yaxis2=**dict(

**titlefont=**dict(

**color=**"#AB5959"

                        ),

**tickfont=**dict(

**color=**"#AB5959"

                        ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.025

                    ),

**yaxis3=**dict(

**titlefont=**dict(

**color=**"#060CBD"

                        ),

**tickfont=**dict(

**color=**"#060CBD"

                        ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.045

                    ),

**yaxis4=**dict(

**titlefont=**dict(

**color=**"#96A35C"

                        ),

**tickfont=**dict(

**color=**"#96A35C"

                        ),

**anchor=**"x",

**overlaying=**"y",

**side=**"right",

                    ),

**yaxis5=**dict(

**titlefont=**dict(

**color=**"#006B33"

                        ),

**tickfont=**dict(

**color=**"#006B33"

                        ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=** 0.97

                    ),

**yaxis6=**dict(

**titlefont=**dict(

**color=**"#98FFFD"

                        ),

**tickfont=**dict(

**color=**"#98FFFD"

                        ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=**0.99

                    )

                )

*# Update layout properties*

                fig.update\_layout(

**title\_text=**"Gas Lock Feature",

**width=**100000,

                )

                st.write(fig)

*# 3 COLUMNS 1*

*# create three columns*

                c1, c2**=** st.columns((10**/**2, 10**/**2))

**with** c1:

*#GRAPH FULL 2*

                    st.markdown("# NORMAL")

                    fig **=** go.Figure()

*#if espjc["State"][q] == "Gas Lock Detected":*

**if** espjc["State"][q] **==** "Normal Condition":

                        e **+=** 1

                        print(espn)

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Discharge Pressure(psi)"][d:e],

**name=**"Discharge Pressure(psi)", **mode=**"lines", **line\_color=**"#FF00EE"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Average Amps(Amps)"][d:e],

**name=**"Average Amps(Amps)",

**yaxis=**"y2", **mode=**"lines", **line\_color=**"#AB5959"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Intake Temperature(F)"][d:e],

**name=**"Intake Temperature(F)",

**yaxis=**"y3", **mode=**"lines", **line\_color=**"#060CBD"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Drive Frequency(Hz)"][d:e],

**name=**"Drive Frequency(Hz)",

**yaxis=**"y4", **mode=**"lines", **line\_color=**"#96A35C"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Motor Temperature(F)"][d:e],

**name=**"Motor Temperature(F)",

**yaxis=**"y5", **mode=**"lines", **line\_color=**"#006B33"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espn["timestamp"][d:e],

**y=**espn["Intake Pressure(psi)"][d:e],

**name=**"Intake Pressure(psi)",

**yaxis=**"y6", **mode=**"lines", **line\_color=**"#98FFFD"

                    ))

*# Create axis objects*

                    fig.update\_layout(

**xaxis=**dict(

**domain=**[0.075, 0.95]

                        ),

**autosize=**False,

**width=**3000,

**height=**2500,

**yaxis=**dict(

**titlefont=**dict(

**color=**"#FF00EE"

                            ),

**tickfont=**dict(

**color=**"#FF00EE"

                            )

                        ),

**yaxis2=**dict(

**titlefont=**dict(

**color=**"#AB5959"

                            ),

**tickfont=**dict(

**color=**"#AB5959"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.025

                        ),

**yaxis3=**dict(

**titlefont=**dict(

**color=**"#060CBD"

                            ),

**tickfont=**dict(

**color=**"#060CBD"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.045

                        ),

**yaxis4=**dict(

**titlefont=**dict(

**color=**"#96A35C"

                            ),

**tickfont=**dict(

**color=**"#96A35C"

                            ),

**anchor=**"x",

**overlaying=**"y",

**side=**"right",

                        ),

**yaxis5=**dict(

**titlefont=**dict(

**color=**"#006B33"

                            ),

**tickfont=**dict(

**color=**"#006B33"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=** 0.97

                        ),

**yaxis6=**dict(

**titlefont=**dict(

**color=**"#98FFFD"

                            ),

**tickfont=**dict(

**color=**"#98FFFD"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=**0.99

                        )

                    )

*# Update layout properties*

                    fig.update\_layout(

**title\_text=**"Gas Lock Feature",

**width=**100000,

                    )

                    st.write(fig)

**with** c2:

*#GRAPH FULL 3*

                    st.markdown("# GAS LOCK")

                    fig **=** go.Figure()

*#if espjc["State"][q] == "Gas Lock Detected":*

**if** espjc["State"][q] **==** "Gas Lock Detected":

                        f **+=** 1

                        print(espg)

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Discharge Pressure(psi)"][d:e],

**name=**"Discharge Pressure(psi)", **mode=**"lines", **line\_color=**"#FF00EE"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Average Amps(Amps)"][d:e],

**name=**"Average Amps(Amps)",

**yaxis=**"y2", **mode=**"lines", **line\_color=**"#AB5959"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Intake Temperature(F)"][d:e],

**name=**"Intake Temperature(F)",

**yaxis=**"y3", **mode=**"lines", **line\_color=**"#060CBD"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Drive Frequency(Hz)"][d:e],

**name=**"Drive Frequency(Hz)",

**yaxis=**"y4", **mode=**"lines", **line\_color=**"#96A35C"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Motor Temperature(F)"][d:e],

**name=**"Motor Temperature(F)",

**yaxis=**"y5", **mode=**"lines", **line\_color=**"#006B33"

                    ))

                    fig.add\_trace(go.Scatter(

**x=**espg["timestamp"][d:e],

**y=**espg["Intake Pressure(psi)"][d:e],

**name=**"Intake Pressure(psi)",

**yaxis=**"y6", **mode=**"lines", **line\_color=**"#98FFFD"

                    ))

*# Create axis objects*

                    fig.update\_layout(

**xaxis=**dict(

**domain=**[0.075, 0.95]

                        ),

**autosize=**False,

**width=**3000,

**height=**2500,

**yaxis=**dict(

**titlefont=**dict(

**color=**"#FF00EE"

                            ),

**tickfont=**dict(

**color=**"#FF00EE"

                            )

                        ),

**yaxis2=**dict(

**titlefont=**dict(

**color=**"#AB5959"

                            ),

**tickfont=**dict(

**color=**"#AB5959"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.025

                        ),

**yaxis3=**dict(

**titlefont=**dict(

**color=**"#060CBD"

                            ),

**tickfont=**dict(

**color=**"#060CBD"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"left",

**position=**0.045

                        ),

**yaxis4=**dict(

**titlefont=**dict(

**color=**"#96A35C"

                            ),

**tickfont=**dict(

**color=**"#96A35C"

                            ),

**anchor=**"x",

**overlaying=**"y",

**side=**"right",

                        ),

**yaxis5=**dict(

**titlefont=**dict(

**color=**"#006B33"

                            ),

**tickfont=**dict(

**color=**"#006B33"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=** 0.97

                        ),

**yaxis6=**dict(

**titlefont=**dict(

**color=**"#98FFFD"

                            ),

**tickfont=**dict(

**color=**"#98FFFD"

                            ),

**anchor=**"free",

**overlaying=**"y",

**side=**"right",

**position=**0.99

                        )

                    )

*# Update layout properties*

                    fig.update\_layout(

**title\_text=**"Gas Lock Feature",

**width=**100000,

                    )

                    st.write(fig)

*# 3 COLUMNS 1*

*# create three columns*

                c1, c2, c3 **=** st.columns((10**/**3, 10**/**3, 10**/**3))

**with** c1:

                    st.markdown('### Disch Pressure(psi)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Discharge Pressure(psi)"][h:m],

**name=**"Discharge Pressure(psi)", **mode=**"lines", **line\_color=**"#FF00EE"

                    ))

                    st.write(fig)

**with** c2:

                    st.markdown('### Average Amps(Amps)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Average Amps(Amps)"][h:m],

**name=**"Discharge Pressure(psi)", **mode=**"lines", **line\_color=**"#AB5959"

                    ))

                    st.write(fig)

**with** c3:

                    st.markdown('### Intake Temperature(F)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Intake Temperature(F)"][h:m],

**name=**"Intake Temperature(F)", **mode=**"lines", **line\_color=**"#060CBD"

                    ))

                    st.write(fig)

*# 3 COLUMNS 2*

*# create three columns*

                c1, c2, c3 **=** st.columns((10**/**3, 10**/**3, 10**/**3))

**with** c1:

                    st.markdown('### Drive Frequency(Hz)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Drive Frequency(Hz)"][h:m],

**name=**"Drive Frequency(Hz)", **mode=**"lines", **line\_color=**"#96A35C"

                    ))

                    st.write(fig)

**with** c2:

                    st.markdown('### Motor Temperature(F)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Motor Temperature(F)"][h:m],

**name=**"Motor Temperature(F)", **mode=**"lines", **line\_color=**"#006B33"

                    ))

                    st.write(fig)

**with** c3:

                    st.markdown('### Intake Pressure(psi)')

                    fig **=** go.Figure()

                    fig.add\_trace(go.Scatter(

**x=**esp["timestamp"][h:m],

**y=**esp["Intake Pressure(psi)"][h:m],

**name=**"Intake Pressure(psi)", **mode=**"lines", **line\_color=**"#98FFFD"

                    ))

                    st.write(fig)

                st.markdown("### Detailed Data View")

*#st.dataframe(esp2[h-1:m-1])*

                st.dataframe(esp4[h**-**1:m**-**1])

                time.sleep(0.1)

            q **+=** 1

        a**+=**11; b**+=**11

    esp4.to\_csv("espdaribaca.csv")