ChatGPT 5.2

CRM_ERP_DASHBOARD.py
Python

Explain this python dash code in detail line by line from scratch in a detailed way

You uploaded a full Dash dashboard Python script. I'll walk you through it **from absolute beginner level**, explaining:

- ✔ What each import does
- ✔ How data is loaded from Excel
- ✔ How cleaning works
- ✔ How Dash layout is built
- ✔ How callbacks update charts
- ✔ How graphs & KPIs are created

I'll go **line by line in logical blocks** so it's easy to understand.

(Source code referenced from your uploaded file CRM_ERP_DASHBOARD)

🧱 PART 1 — BASIC FILE INFO

python Copy code

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
```

👉 What this means:

• Tells computer to run using Python 3
• Supports UTF-8 characters (emojis, symbols, etc.)

Not required but good practice.

---

python Copy code

```
"""
Full Management-Ready Dash Dashboard for Leads ...
"""
```

👉 This is a comment block

Just describes what this program does.

(No effect on code execution)

📦 PART 2 — IMPORTING LIBRARIES

python Copy code

```
import os
```

👉 Used to:

- Work with folders
- Find files
- Join paths

Example: finding Excel files in a folder

---

python Copy code

```
import pandas as pd
```

👉 Pandas = main data handling tool

Used for:

- Reading Excel
- Filtering data
- Grouping data

Think of pandas as **Excel inside Python**

---

python Copy code

```python
from datetime import datetime
```

👉 Used for:

• Handling dates
• Comparing due dates
• Creating trends

---

python Copy code

```python
import plotly.graph_objects as go
import plotly.express as px
```

👉 Plotly = chart library

- `px` = quick charts (pie, bar, line)
- `go` = advanced custom charts (heatmaps, multi lines)

---

python Copy code

```python
import dash
from dash import dcc, html, Input, Output, dash_table
```

👉 Dash = web dashboard framework

Item	Meaning
html	HTML components
dcc	charts, dropdowns
Input	receives user selection

Item	Meaning
Output	updates charts
dash_table	table view

---

python Copy code

```
import dash_bootstrap_components as dbc
```

👉 Adds **Bootstrap styling**

Makes dashboard look professional.

📁 PART 3 — SETTING DATA PATH

python Copy code

```
BASE_DIR = os.path.dirname(os.path.abspath(__file__))
```

👉 Finds:

📁 Folder where this Python file exists

---

python Copy code

```
DATA_FOLDER = os.path.join(BASE_DIR, "Leads_Data")
```

👉 Creates path like:

markdown Copy code

```
your_project/
    CRM_ERP_DASHBOARD.py
    Leads_Data/
```

```
file1.xlsx
file2.xlsx
```

So program reads Excel from `Leads_Data` folder.

📊 PART 4 — LOADING ALL EXCEL FILES

python Copy code

```python
def load_all_excels(folder):
```

👉 Defines a function (reusable code)

Purpose:
👉 Load ALL Excel files inside folder

---

python Copy code

```python
dfs = []
```

👉 Empty list

Will store many Excel DataFrames.

---

python Copy code

```python
for file in os.listdir(folder):
```

👉 Loops through each file in folder

---

python Copy code

```python
if file.startswith("~$"):
```

```
    continue
```

👉 Skips temp Excel files

(Excel creates these automatically)

---

python Copy code

```
path = os.path.join(folder, file)
```

👉 Makes full file path

Example:

Copy code

```
Leads_Data/January.xlsx
```

---

python Copy code

```
if file.endswith(".xlsx"):
    df = pd.read_excel(path, engine="openpyxl")
elif file.endswith(".xls"):
    df = pd.read_excel(path, engine="xlrd")
```

👉 Reads Excel into pandas table

Handles:

- new format (.xlsx)
- old format (.xls)

---

python Copy code

```
df["Source_File"] = file
```

👉 Adds column showing:

Which Excel file data came from

Useful for tracking.

---

python Copy code

```
dfs.append(df)
```

👉 Adds this table into list

---

python Copy code

```
return pd.concat(dfs, ignore_index=True)
```

👉 Combines all Excel tables into ONE big table

Like stacking sheets vertically.

---

python Copy code

```
df_raw = load_all_excels(DATA_FOLDER)
```

👉 Calls the function

👉 Loads all Excel into `df_raw`

🧹 PART 5 — DATA CLEANING

python Copy code

```
df = df_raw.copy()
```

👉 Makes safe copy of original data

---

python Copy code

```
df.columns = df.columns.str.strip().str.replace(" ", "_").str.replace("/", "_")
```

👉 Cleans column names:

Before	After
Lead Id	Lead_Id
Due date	Due_date

Why?

Python likes clean names.

---

python Copy code

```
df["Value_in_Rs_(Lakhs)"] = pd.to_numeric(...).fillna(0)
```

👉 Converts money column into numbers

• removes text errors
• fills empty with 0

---

python Copy code

```
df[col] = pd.to_datetime(df[col], errors="coerce")
```

👉 Converts date columns into date format

If invalid → becomes empty (NaT)

---

python Copy code

```
df["Year"] = df["Creation_Date"].dt.year
```

👉 Extracts year:

2023, 2024, etc.

Used for yearly trend.

---

python Copy code

```
df["Sector"] = df["Lead_Type"].apply(...)
```

👉 If Lead_Type contains "defence" → Defence
Else → Civil

Creates new column.

---

python Copy code

```
df["Is_Overdue"] = df["Due_date"] < today
```

👉 Checks:

If due date passed → overdue = True

🌐 PART 6 — CREATE DASH APP

python Copy code

```
app = dash.Dash(__name__, external_stylesheets=[dbc.themes.BOOTSTRAP])
```

👉 Starts Dash web app

👉 Uses Bootstrap theme

python Copy code

```
app.title = "Management Lead Dashboard"
```

👉 Browser tab name

🧱 PART 7 — DASHBOARD LAYOUT

This part builds the **web page**

---

python Copy code

```
dbc.Container(fluid=True, children=[ ... ])
```

👉 Main wrapper of page

fluid=True → full width

Title:

python Copy code

```
html.H2("📊 Management Lead Dashboard")
```

👉 Big heading

Dropdown Filters:

python Copy code

```
dcc.Dropdown(...)
```

Each dropdown:

- UNIT/SBU
- Customer
- Year
- Lead Type
- Lead Stage
- Lead Status

They:

✔ show unique values
✔ allow multi select

Buttons + Collapse

Example:

python Copy code

```
dbc.Button("Show / Hide Executive KPIs")
dbc.Collapse(...)
```

👉 Button toggles visibility

👉 Collapse hides/shows content

Used for:

- KPIs
- Charts
- Tables

Graph placeholders:

python Copy code

```
dcc.Graph(id="lead_status_pie")
```

👉 Empty space where chart appears

Raw table:

python Copy code

```
dash_table.DataTable(...)
```

👉 Shows full data like Excel

With:

• filter
• sort
• pagination

🔁 PART 8 — TOGGLE CALLBACK LOOP

python Copy code

```
for section in ["kpi", "lead_sector", ...]:
```

👉 Creates same toggle logic for each section

---

python Copy code

```
@app.callback(...)
```

👉 Dash callback:

When button clicked → hide/show section

---

python Copy code

```
return not is_open
```

👉 Switches visibility

📈 PART 9 — MAIN DASH CALLBACK

This is the **heart of dashboard**

---

python Copy code

```
@app.callback(
    Output(...),
    Input(...)
)
```

Means:

When user changes:

• dropdowns

Update:

• KPIs
• charts
• table

---

python Copy code

```
def update_dashboard(...):
```

👉 Runs every time filter changes

Copy dataframe:

python Copy code

```
dff = df.copy()
```

Safe working copy.

Date conversion again (safety)

Ensures all date columns valid.

Normalize status text:

python Copy code

```
def normalize_assignment_status(x):
```

👉 Cleans messy text like:

"approved by admin" → Approved

Standardizes values.

Apply filters:

python Copy code

```
if sbu: dff = dff[dff["UNIT_SBU"].isin(sbu)]
```

👉 If user selects SBU → keep only those rows

Same for all filters.

📊 PART 10 — KPI CALCULATIONS

Examples:

python Copy code

```
total_leads = dff["Lead_Id"].nunique()
```

👉 Counts unique leads

---

python Copy code

```
total_value = sum money
```

👉 Total business value

---

python Copy code

```
top_owner = most leads owner
```

👉 Finds who owns most leads

---

python Copy code

```
single_vendor, multi_vendor
```

👉 Counts vendor type

Reasons grouping:

Groups lead IDs by approval reason.

Creates detailed info cards.

📦 PART 11 — KPI CARDS

python Copy code

```
dbc.Card(...)
```

👉 Each KPI shown in a nice box:

- Total leads
- Total value
- Top owner
- Top SBU
- Vendors
- Approved reasons
- Non approved reasons

📉 PART 12 — CHARTS

Pie:

python Copy code

```
px.pie(dff, names="Lead_Status")
```

Lead status distribution.

Bar:

python Copy code

```
px.bar(grouped data)
```

SBU vs Value

Owner vs Count

Line:

python Copy code

```
px.line(yearly data)
```

Year wise trend.

Financial summary:

Planned vs Achieved vs Lost money.

Date trend:

Uses `go.Scatter` to plot:

- Tech bid date
- Commercial bid date
- PNC date

Multiple lines in one chart.

Risk Heatmap:

python Copy code

```
go.Heatmap(...)
```

Shows overdue leads by:

SBU × Lead status

Red = high risk

Top 10 leads:

Sorts by value and plots bar chart.

📋 PART 13 — RAW TABLE

python Copy code

```
data = dff.to_dict("records")
columns = [{"name": c, "id": c}]
```

👉 Converts pandas → Dash table format

▶️ PART 14 — RUN SERVER

python Copy code

```python
if __name__ == "__main__":
    app.run_server(...)
```

👉 Starts local web server

You open in browser:

arduino Copy code

```arduino
http://localhost:8001
```

✅ SUMMARY IN SIMPLE WORDS

This code:

- ✔ Loads many Excel files
- ✔ Cleans data
- ✔ Builds web dashboard
- ✔ Adds filters
- ✔ Shows KPIs
- ✔ Creates many charts
- ✔ Shows raw table
- ✔ Updates everything dynamically

If you'd like, I can next:

- ✅ Explain Dash callbacks with simple diagram
- ✅ Explain how dropdown connects to chart

✅ Rewrite a smaller version for practice

✅ Draw flow of data step-by-step

Just tell me 👍

Sources