**Technical Documentation**

**Project Title:** Multi-Source Sales Data Pipeline & Business Insights Dashboard

**Intern:** Alahari Bhargavi  
**Company:** Flipkart Pvt Ltd  
**Mentor:** Jayanth Pathwari  
**Timeline:** 45 Days

**1. Objective**

To build a robust data pipeline that consolidates sales data from multiple formats (CSV, JSON, Excel), cleans it, performs transformations, and delivers business insights through visualizations and PDF reports.

### ****2. Tools and Libraries Used****

* Python 3.13
* pandas, numpy
* matplotlib, seaborn
* openpyxl, json
* tabulate, jinja2, pdfkit

### ****3. Dataset Description****

**Files Provided:**

* sales\_data.csv: Contains daily sales transactions
* product\_metadata.json: Maps products to categories
* region\_info.xlsx: Contains regional manager information

**Sample Preview:**

Date,Product,Units Sold,Revenue,Region

01-01-2025,Widget A,10,100,North

02-01-2025,Widget B,5,75,South

03-01-2025,Widget A,8,80,North

--------------------------------------------------------------------------------------------------------------------------------------

[

{"Product": "Widget A", "Category": "Gadgets"},

{"Product": "Widget B", "Category": "Accessories"}

]

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| |  |  | | --- | --- | | **Region** | **Manager** | | North | Alice | | South | Bob |  ****4. Code Modules******A. data\_integration.py** import pandas as pd  import json  sales\_df = pd.read\_csv("sales\_data.csv")  with open("product\_metadata.json") as f:      product\_data = json.load(f)  product\_df = pd.DataFrame(product\_data)  region\_df = pd.read\_excel("region\_info.xlsx")  merged = pd.merge(sales\_df, product\_df, on="Product", how="left")  final\_df = pd.merge(merged, region\_df, on="Region", how="left")  final\_df.to\_csv("final\_output.csv", index=False)  print(" Data integration completed.")  print(final\_df.head())  Output :  PS C:\Users\admin\Desktop\flipkart> & C:/Users/admin/AppData/Local/Programs/Python/Python313/python.exe c:/Users/admin/Desktop/flipkart/data\_integration.py  Data integration completed.  Date Product Units Sold Revenue Region Category Manager  0 01-01-2025 Widget A 10 100 North Gadgets Alice  1 02-01-2025 Widget B 5 75 South Accessories Bob  2 03-01-2025 Widget A 8 80 North Gadgets Alice |  |
|  |  |

#### **B. data\_cleaning.py**

import pandas as pd

df = pd.read\_csv('final\_output.csv')

print("Missing values before cleaning:")

print(df.isnull().sum())

print()

df['Date'] = pd.to\_datetime(df['Date'], format='%d-%m-%Y', errors='coerce')

df = df.dropna(subset=['Date', 'Product', 'Revenue'])

print("Missing values after cleaning:")

print(df.isnull().sum())

print()

print("Data types after conversion:")

print(df.dtypes)

df.to\_csv('final\_output\_cleaned.csv', index=False)

print("\n Cleaned data saved to 'final\_output\_cleaned.csv'")

Output :

Missing values before cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Missing values after cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Data types after conversion:

Date datetime64[ns]

Product object

Units Sold int64

Revenue int64

Region object

Category object

Manager object

dtype: object

Cleaned data saved to 'final\_output\_cleaned.csv'

#### **C. eda\_visualization.py**

import pandas as pd

import matplotlib.pyplot as plt

df = pd.read\_csv('final\_output\_cleaned.csv')

df['Date'] = pd.to\_datetime(df['Date'], format='%Y-%m-%d')

df['Month'] = df['Date'].dt.to\_period('M')

monthly\_sales = df.groupby('Month')['Revenue'].sum().reset\_index()

plt.figure(figsize=(8, 5))

plt.plot(monthly\_sales['Month'].astype(str), monthly\_sales['Revenue'], marker='o')

plt.title("Monthly Revenue Trend")

plt.xlabel("Month")

plt.ylabel("Total Revenue")

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

region\_sales = df.groupby('Region')['Revenue'].sum().reset\_index()

plt.figure(figsize=(6, 4))

plt.bar(region\_sales['Region'], region\_sales['Revenue'], color='skyblue')

plt.title("Region-wise Revenue")

plt.xlabel("Region")

plt.ylabel("Revenue")

plt.tight\_layout()

plt.show()

category\_sales = df.groupby('Category')['Revenue'].sum().reset\_index()

plt.figure(figsize=(6, 4))

plt.bar(category\_sales['Category'], category\_sales['Revenue'], color='orange')

plt.title("Category-wise Revenue")

plt.xlabel("Category")

plt.ylabel("Revenue")

plt.tight\_layout()

plt.show()

summary = df.groupby(['Region', 'Category'])['Revenue'].sum().reset\_index()

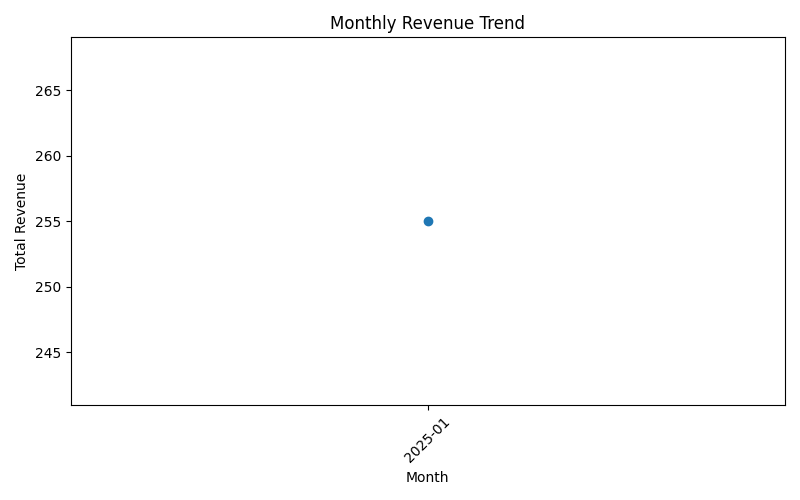
print("\nRevenue by Region and Category:")

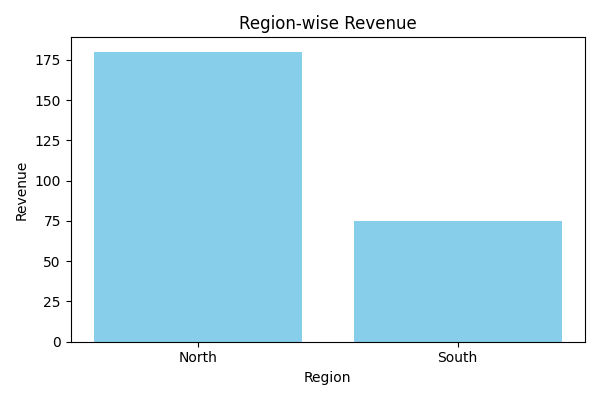
print(summary)

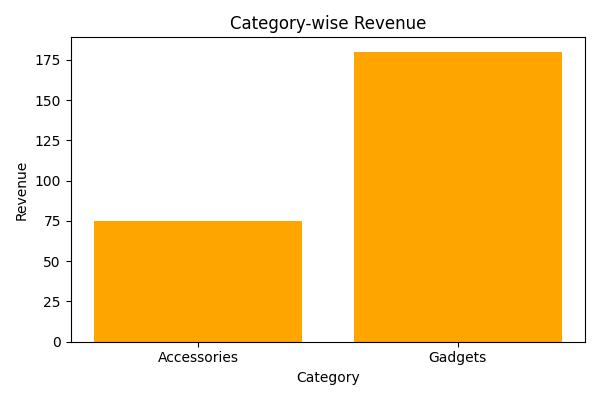
summary.to\_csv('summary.csv', index=False)

print(" Summary exported to summary.csv")

Output :







#### **D. data\_visualization.py**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read\_csv('final\_output\_cleaned.csv')

df['Date'] = pd.to\_datetime(df['Date'], format='%Y-%m-%d')

df['Month'] = df['Date'].dt.to\_period('M')

sns.set(style="whitegrid")

monthly\_sales = df.groupby('Month')['Revenue'].sum().reset\_index()

region\_sales = df.groupby('Region')['Revenue'].sum().reset\_index()

category\_sales = df.groupby('Category')['Revenue'].sum().reset\_index()

fig, axs = plt.subplots(3, 1, figsize=(10, 15))

sns.lineplot(ax=axs[0], x=monthly\_sales['Month'].astype(str), y='Revenue', data=monthly\_sales, marker='o')

axs[0].set\_title("Monthly Revenue Trend")

axs[0].set\_xlabel("Month")

axs[0].set\_ylabel("Total Revenue")

axs[0].tick\_params(axis='x', rotation=45)

sns.barplot(ax=axs[1], x='Region', y='Revenue', data=region\_sales, palette="Blues\_d")

axs[1].set\_title("Region-wise Revenue")

axs[1].set\_xlabel("Region")

axs[1].set\_ylabel("Revenue")

sns.barplot(ax=axs[2], x='Category', y='Revenue', data=category\_sales, palette="Oranges\_d")

axs[2].set\_title("Category-wise Revenue")

axs[2].set\_xlabel("Category")

axs[2].set\_ylabel("Revenue")

plt.tight\_layout()

plt.show()

fig.savefig("dashboard.png", dpi=300)

Output :

#### 

#### **D. generate\_report.py**

import pandas as pd

import pdfkit

from jinja2 import Environment, FileSystemLoader

df = pd.read\_csv('final\_output\_cleaned.csv')

summary = pd.read\_csv('summary.csv')

total\_revenue = df['Revenue'].sum()

top\_region = df.groupby('Region')['Revenue'].sum().idxmax()

top\_category = df.groupby('Category')['Revenue'].sum().idxmax()

env = Environment(loader=FileSystemLoader('.'))

template = env.get\_template("report\_template.html")

html\_out = template.render(

    total\_revenue=total\_revenue,

    top\_region=top\_region,

    top\_category=top\_category,

    summary\_table=summary.to\_html(index=False)

)

config = pdfkit.configuration(wkhtmltopdf=r"C:\Program Files\wkhtmltopdf\bin\wkhtmltopdf.exe")

pdfkit.from\_string(html\_out, "sales\_report.pdf", configuration=config)

print(" PDF report generated: sales\_report.pdf")

Output :

PDF report generated: sales\_report.pdf

#### **E. automate\_pipeline.py**

import argparse

import subprocess

def run\_script(script\_name):

    print(f"\n Running {script\_name}...")

    subprocess.run(["python", script\_name], check=True)

parser = argparse.ArgumentParser(description="Flipkart Data Analysis Pipeline")

parser.add\_argument('--clean', action='store\_true', help='Run data cleaning')

parser.add\_argument('--eda', action='store\_true', help='Run exploratory data analysis')

parser.add\_argument('--report', action='store\_true', help='Generate PDF report')

parser.add\_argument('--all', action='store\_true', help='Run all steps')

args = parser.parse\_args()

if args.clean:

    run\_script("data\_cleaning.py")

if args.eda:

    run\_script("eda\_visualization.py")

if args.report:

    run\_script("generate\_report.py")

if args.all:

    run\_script("data\_integration.py")

    run\_script("data\_cleaning.py")

    run\_script("eda\_visualization.py")

    run\_script("generate\_report.py")

### ****5. Execution Instructions****

python automate\_pipeline.py –all

Make sure to install all dependencies from requirements.txt and have wkhtmltopdf added to system PATH

Output :

Running data\_cleaning.py...

Missing values before cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Missing values after cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Data types after conversion:

Date datetime64[ns]

Product object

Units Sold int64

Revenue int64

Region object

Category object

Manager object

dtype: object

Cleaned data saved to 'final\_output\_cleaned.csv'

C:\Users\admin\Desktop\flipkart>python automate\_pipeline.py --all

Running data\_integration.py...

Data integration completed.

Date Product Units Sold Revenue Region Category Manager

0 01-01-2025 Widget A 10 100 North Gadgets Alice

1 02-01-2025 Widget B 5 75 South Accessories Bob

2 03-01-2025 Widget A 8 80 North Gadgets Alice

Running data\_cleaning.py...

Missing values before cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Missing values after cleaning:

Date 0

Product 0

Units Sold 0

Revenue 0

Region 0

Category 0

Manager 0

dtype: int64

Data types after conversion:

Date datetime64[ns]

Product object

Units Sold int64

Revenue int64

Region object

Category object

Manager object

dtype: object

Cleaned data saved to 'final\_output\_cleaned.csv'

Running eda\_visualization.py...

EDA visualizations saved as PNGs.

Running generate\_report.py...

PDF report generated: sales\_report.pdf

### ****6. Login Dashboard****

**Objective:**  
Develop a secure, user-accessible web interface using Flask to manage access to sales reports (e.g., PDF report generated in previous steps).

**Tools Used:**

* Python
* Flask
* HTML/CSS (Jinja2 templates)

**Features:**

* Secure login authentication
* Dashboard view to download sales\_report.pdf

### app.py

from flask import Flask, render\_template, request, redirect, url\_for, send\_file

app = Flask(\_\_name\_\_)

USERNAME = "admin"

PASSWORD = "flipkart123"

@app.route('/', methods=['GET', 'POST'])

def login():

    error = None

    if request.method == 'POST':

        if request.form['username'] == USERNAME and request.form['password'] == PASSWORD:

            return redirect(url\_for('dashboard'))

        else:

            error = 'Invalid username or password'

    return render\_template('login.html', error=error)

@app.route('/dashboard')

def dashboard():

    return render\_template('dashboard.html')

@app.route('/download')

def download\_report():

    return send\_file('sales\_report.pdf', as\_attachment=True)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)

### Output :

\* Serving Flask app 'app'

\* Debug mode: on

WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.

\* Running on http://127.0.0.1:5000

Press CTRL+C to quit

\* Restarting with stat

\* Debugger is active!

\* Debugger PIN: 957-077-339

Login.html

<!DOCTYPE html>

<html>

<head>

    <title>Login</title>

</head>

<body>

    <h2>Login to Dashboard</h2>

    {% if error %}

        <p style="color:red;">{{ error }}</p>

    {% endif %}

    <form method="POST">

        <input type="text" name="username" placeholder="Username" required><br><br>

        <input type="password" name="password" placeholder="Password" required><br><br>

        <input type="submit" value="Login">

    </form>

</body>

</html>

Dashboard.html

<!DOCTYPE html>

<html>

<head>

    <title>Dashboard</title>

</head>

<body>

    <h2>Welcome to Flipkart Dashboard</h2>

    <p><a href="/download">Download PDF Report</a></p>

</body>

</html>

report\_template.html

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8">

    <title>Sales Report</title>

    <style>

        body { font-family: Arial; padding: 20px; }

        h1 { color: #2c3e50; }

        table { width: 100%; border-collapse: collapse; margin-top: 20px; }

        th, td { padding: 8px 12px; border: 1px solid #ccc; text-align: center; }

    </style>

</head>

<body>

    <h1>Flipkart Sales Report</h1>

    <p><strong>Total Revenue:</strong> ₹{{ total\_revenue | round(2) }}</p>

    <p><strong>Top Performing Region:</strong> {{ top\_region }}</p>

    <p><strong>Top Product Category:</strong> {{ top\_category }}</p>

    <h2>Revenue Summary by Region and Category</h2>

    {{ summary\_table | safe }}

</body>

</html>

Total Output :

Flipkart Sales Report Total Revenue: ₹255 Top Performing Region: North Top Product Category: Gadgets Revenue Summary by Region and Category Region Category Revenue North Gadgets 180 South Accessories 75

Project Structure :

flipkart/

│

├── data\_integration.py # Step 1: Merges CSV, JSON, Excel into one DataFrame

├── data\_cleaning.py # Step 2: Cleans data (missing values, date format)

├── eda\_visualization.py # Step 3: Generates charts with matplotlib/seaborn

├── generate\_report.py # Step 4: Converts summary to PDF

├── automate\_pipeline.py # Step 5: Automates all scripts using CLI options

│

├── sales\_data.csv # Raw CSV data (sales)

├── product\_metadata.json # Raw JSON data (product categories)

├── region\_info.xlsx # Raw Excel data (region managers)

│

├── final\_output.csv # Merged data after integration

├── final\_output\_cleaned.csv # Cleaned version of the merged data

│

├── monthly\_revenue.png # EDA Visualization 1

├── region\_revenue.png # EDA Visualization 2

├── category\_revenue.png # EDA Visualization 3

│

├── sales\_report.pdf # Final PDF business report

│

├── requirements.txt # All required Python libraries

├── README.md # GitHub instructions (to be written)

│

├── app.py # Flask-based login dashboard

│

├── templates/ # HTML templates for Flask

│ ├── login.html # Login form UI

│ └── dashboard.html # Jinja2 HTML used for PDF export

│

└── static/ # Reserved for future use (CSS, JS, image)

└── report\_template.html