Project Name - PhonePe Transaction Insights

Project Type - Data Visualization / Dashboard Project

Contribution - Individual

Name - Khushi vyas

Project Summary:

This project, *"PhonePe Pulse Data Analysis using Python, MySQL, and Streamlit"*, focuses on analyzing India's digital payment trends using real data released by **PhonePe**, one of the country's largest fintech platforms. The goal of this project is to extract valuable insights about transaction patterns, user behavior, and growth in digital payments across Indian states.

The dataset used in this project is derived from the **PhonePe Pulse GitHub repository**, which contains aggregated data on transactions and users across different states and years. The data was originally in JSON format, which was read and loaded into a **MySQL database** using Python.

After successful data storage, a **Streamlit-based interactive dashboard** was developed to visualize key insights. The dashboard displays:

- State-wise and year-wise transaction counts and amounts,
- Comparison of digital payment categories like recharge, bill payments, peer-to-peer transfers, etc.,
- Growth trends in transaction volume and value across time.

By combining **Python (for data processing)**, **MySQL (for structured storage)**, and **Streamlit (for visualization)**, this project demonstrates how data science techniques can be applied to real-world industry data. It highlights the increasing adoption of digital payments in India and provides an analytical view of the fintech landscape.

This industry-based project not only enhances understanding of data analysis tools but also offers hands-on experience with real business data, bridging the gap between academic learning and real-world data analytics applications.


PhonePe
INDIA'S PAYMENTS APP

Code of this project:

```
import mysql.connector

conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Khushi@0054",
    database="phonepe_db"
)

if conn.is_connected():
    print("Database Connected Successfully!")
else:
    print("Connection Failed")

import mysql.connector
import json
import os

# Step 1: Database Connection
conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Khushi@0054",    # apna password
    database="phonepe_db"      # apna database
)
cursor = conn.cursor()
print("Database Connected Successfully!")

# Step 2: Path to data
DATA_PATH = r"D:\labmentic project 2\pulse-master\pulse-
master\data\aggregated\transaction\country\india\state"

# Step 3: Loop through states, years, and files
insert_query = """
INSERT INTO aggregated_transaction (state, year, name, count, amount)
VALUES (%s, %s, %s, %s, %s)
"""

row_count = 0
```

```
for state in os.listdir(DATA_PATH):
    state_path = os.path.join(DATA_PATH, state)
    if not os.path.isdir(state_path):
        continue

    for year in os.listdir(state_path):
        year_path = os.path.join(state_path, year)
        if not os.path.isdir(year_path):
            continue

        for file in os.listdir(year_path):
            if file.endswith('.json'):
                print(f"Processing: {state} | {year} | {file}")  # 👈 progress
print
                file_path = os.path.join(year_path, file)

                with open(file_path, 'r', encoding='utf-8') as f:
                    data = json.load(f)

                    # Check data structure
                    if data.get('data') and data['data'].get('transactionData'):
                        for record in data['data']['transactionData']:
                            name = record['name']
                            count = record['paymentInstruments'][0]['count']
                            amount = record['paymentInstruments'][0]['amount']

                            cursor.execute(insert_query, (state, year, name,
count, amount))
                            row_count += 1

# Step 4: Commit and Close
conn.commit()
print(f"Data inserted successfully! Total rows: {row_count}")

cursor.close()
conn.close()
```

Output of this code:

Processing: tripura | 2021 | 4.json

Processing: tripura | 2022 | 1.json

Processing: tripura | 2022 | 2.json

Processing: tripura | 2022 | 3.json

Processing: tripura | 2022 | 4.json

Processing: tripura | 2023 | 1.json

Processing: tripura | 2023 | 2.json

Processing: tripura | 2023 | 3.json

Processing: tripura | 2023 | 4.json

Processing: tripura | 2024 | 1.json

Processing: tripura | 2024 | 2.json

Processing: tripura | 2024 | 3.json

Processing: tripura | 2024 | 4.json

Processing: uttar-pradesh | 2018 | 1.json

Processing: uttar-pradesh | 2018 | 2.json

Processing: uttar-pradesh | 2018 | 3.json

Processing: uttar-pradesh | 2018 | 4.json

Processing: uttar-pradesh | 2019 | 1.json

Processing: uttar-pradesh | 2019 | 2.json

Processing: uttar-pradesh | 2019 | 3.json

Processing: uttar-pradesh | 2019 | 4.json

Processing: uttar-pradesh | 2020 | 1.json

Processing: uttar-pradesh | 2020 | 2.json

Processing: uttar-pradesh | 2020 | 3.json

Processing: uttar-pradesh | 2020 | 4.json

Processing: uttar-pradesh | 2021 | 1.json

Processing: uttar-pradesh | 2021 | 2.json

Processing: uttar-pradesh | 2021 | 3.json

Processing: uttar-pradesh | 2021 | 4.json

Processing: uttar-pradesh | 2022 | 1.json

Processing: uttar-pradesh | 2022 | 2.json

Processing: uttar-pradesh | 2022 | 3.json

Processing: uttar-pradesh | 2022 | 4.json

Processing: uttar-pradesh | 2023 | 1.json

Processing: uttar-pradesh | 2023 | 2.json

Processing: uttar-pradesh | 2023 | 3.json

Processing: uttar-pradesh | 2023 | 4.json

Processing: uttar-pradesh | 2024 | 1.json

Processing: uttar-pradesh | 2024 | 2.json

Processing: uttar-pradesh | 2024 | 3.json

Processing: uttar-pradesh | 2024 | 4.json

Processing: uttarakhand | 2018 | 1.json

Processing: uttarakhand | 2018 | 2.json

Processing: uttarakhand | 2018 | 3.json

Processing: uttarakhand | 2018 | 4.json

Processing: uttarakhand | 2019 | 1.json

Processing: uttarakhand | 2019 | 2.json

Processing: uttarakhand | 2019 | 3.json

Processing: uttarakhand | 2019 | 4.json

Processing: uttarakhand | 2020 | 1.json

Processing: uttarakhand | 2020 | 2.json

Processing: uttarakhand | 2020 | 3.json

Processing: uttarakhand | 2020 | 4.json

Processing: uttarakhand | 2021 | 1.json

Processing: uttarakhand | 2021 | 2.json

Processing: uttarakhand | 2021 | 3.json

Processing: uttarakhand | 2021 | 4.json

Processing: uttarakhand | 2022 | 1.json

Processing: uttarakhand | 2022 | 2.json

Processing: uttarakhand | 2022 | 3.json

Processing: uttarakhand | 2022 | 4.json

Processing: uttarakhand | 2023 | 1.json

Processing: uttarakhand | 2023 | 2.json

Processing: uttarakhand | 2023 | 3.json

Processing: uttarakhand | 2023 | 4.json

Processing: uttarakhand | 2024 | 1.json

Processing: uttarakhand | 2024 | 2.json

Processing: uttarakhand | 2024 | 3.json

Processing: uttarakhand | 2024 | 4.json

Processing: west-bengal | 2018 | 1.json

Processing: west-bengal | 2018 | 2.json

Processing: west-bengal | 2018 | 3.json

Processing: west-bengal | 2018 | 4.json

Processing: west-bengal | 2019 | 1.json

Processing: west-bengal | 2019 | 2.json

Processing: west-bengal | 2019 | 3.json

Processing: west-bengal | 2019 | 4.json

Processing: west-bengal | 2020 | 1.json

Processing: west-bengal | 2020 | 2.json

Processing: west-bengal | 2020 | 3.json

Processing: west-bengal | 2020 | 4.json

Processing: west-bengal | 2021 | 1.json

Processing: west-bengal | 2021 | 2.json

Processing: west-bengal | 2021 | 3.json

Processing: west-bengal | 2021 | 4.json

Processing: west-bengal | 2022 | 1.json

Processing: west-bengal | 2022 | 2.json

Processing: west-bengal | 2022 | 3.json

Processing: west-bengal | 2022 | 4.json

Processing: west-bengal | 2023 | 1.json

Processing: west-bengal | 2023 | 2.json

Processing: west-bengal | 2023 | 3.json

Processing: west-bengal | 2023 | 4.json

Processing: west-bengal | 2024 | 1.json

Processing: west-bengal | 2024 | 2.json

Processing: west-bengal | 2024 | 3.json

Processing: west-bengal | 2024 | 4.json

Data inserted successfully! Total rows: 5034

PS D:\New folder>

* History restored

Analyze data:

```
import pandas as pd
import mysql.connector
import matplotlib.pyplot as plt

# Step 1: Connect to database
conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="Khushi@0054",
    database="phonepe_db"
)

# Step 2: Read data
query = "SELECT * FROM aggregated_transaction;"
df = pd.read_sql(query, conn)
print("Data loaded from MySQL successfully!")
print(df.head())

# Step 3: Top 10 states by transaction amount
top_states =
df.groupby("state")["amount"].sum().sort_values(ascending=False).head(10)
plt.figure(figsize=(10,6))
top_states.plot(kind="bar", color="skyblue")
plt.title("Top 10 States by Transaction Amount")
plt.xlabel("State")
plt.ylabel("Total Amount (in ₹)")
plt.tight_layout()
plt.show()

# Step 4: Yearly trend
df["year"] = df["year"].astype(int)
yearly_trend = df.groupby("year")["amount"].sum()
plt.figure(figsize=(8,5))
yearly_trend.plot(marker="o", color="green")
plt.title("Yearly Transaction Growth (All India)")
plt.xlabel("Year")
plt.ylabel("Total Amount (in ₹)")
plt.grid(True)
plt.show()
```

Output:

PS D:\New folder> & "C:/Users/khushi vyas/AppData/Local/Programs/Python/Python313/python.exe" "d:/New folder/analyze_data.py"

d:\New folder\analyze_data.py:15: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database

y"

d:\New folder\analyze_data.py:15: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database

d:\New folder\analyze_data.py:15: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database

string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.

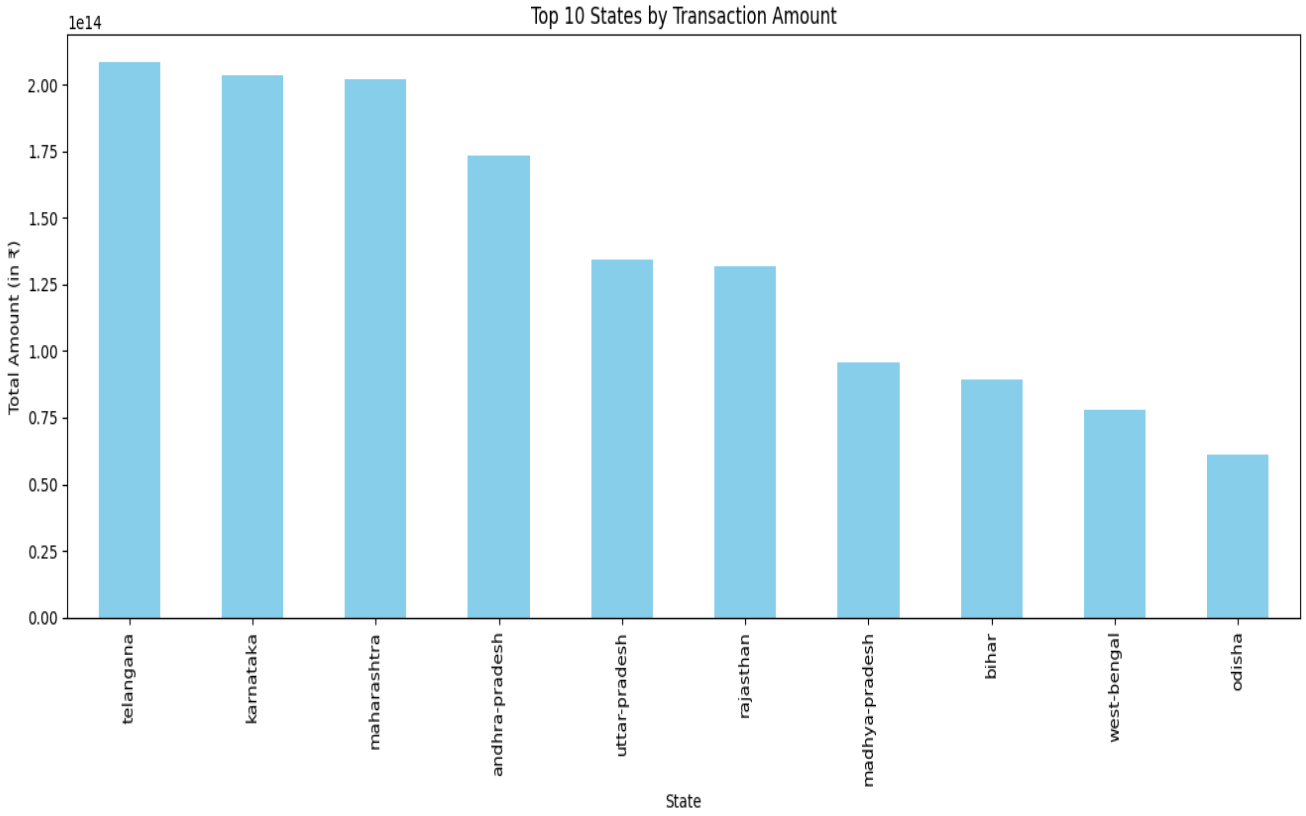
df = pd.read_sql(query, conn)

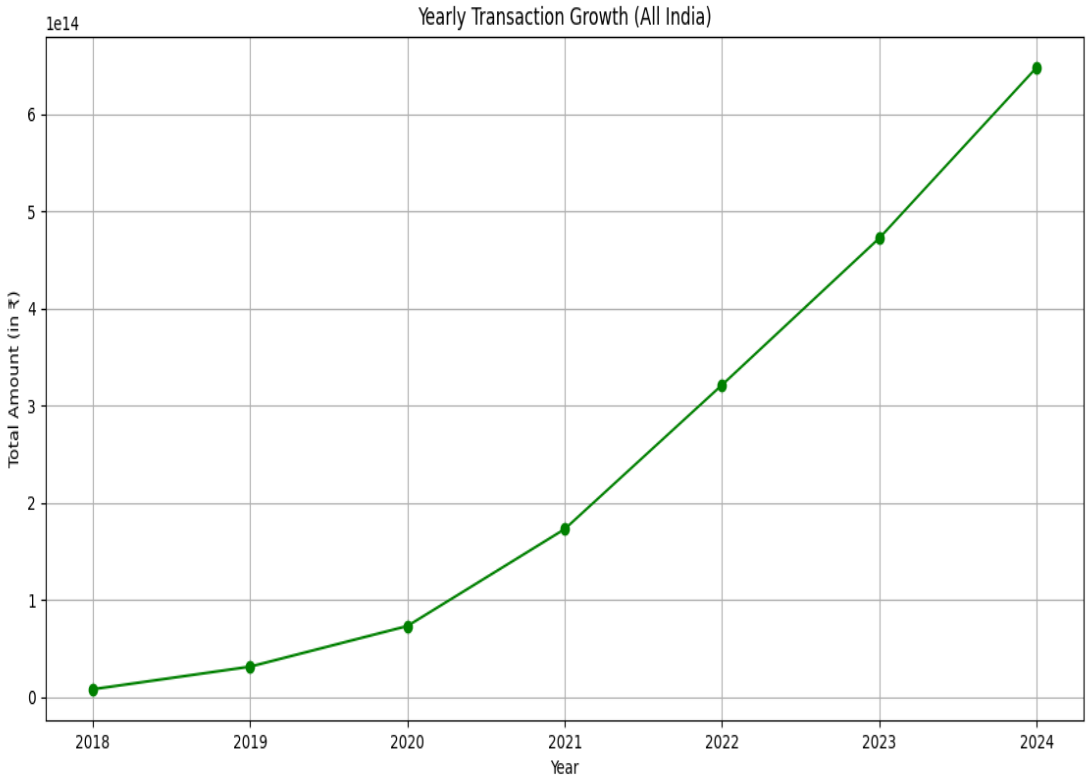
Data loaded from MySQL successfully!

	id	state	year	name	count	amount
0	1	andaman-&-nicobar-islands	2018	Recharge & bill payments	4200	1.845307e+06
1	2	andaman-&-nicobar-islands	2018	Peer-to-peer payments	1871	1.213866e+07

	id	state	year	name	count	amount
0	1	andaman-&-nicobar-islands	2018	Recharge & bill payments	4200	1.845307e+06
1	2	andaman-&-nicobar-islands	2018	Peer-to-peer payments	1871	1.213866e+07
1	2	andaman-&-nicobar-islands	2018	Peer-to-peer payments	1871	1.213866e+07
2	3	andaman-&-nicobar-islands	2018	Merchant payments	298	4.525072e+05
3	4	andaman-&-nicobar-islands	2018	Financial Services	33	1.060142e+04
4	5	andaman-&-nicobar-islands	2018	Others	256	1.846899e+05

Visuliaztion :





(x, y) = (2021.991, 5.76e+14)

Output:
PS D:\New folder> streamlit run "d:/New folder/app.py"

>> >>

You can now view your Streamlit app in your browser.

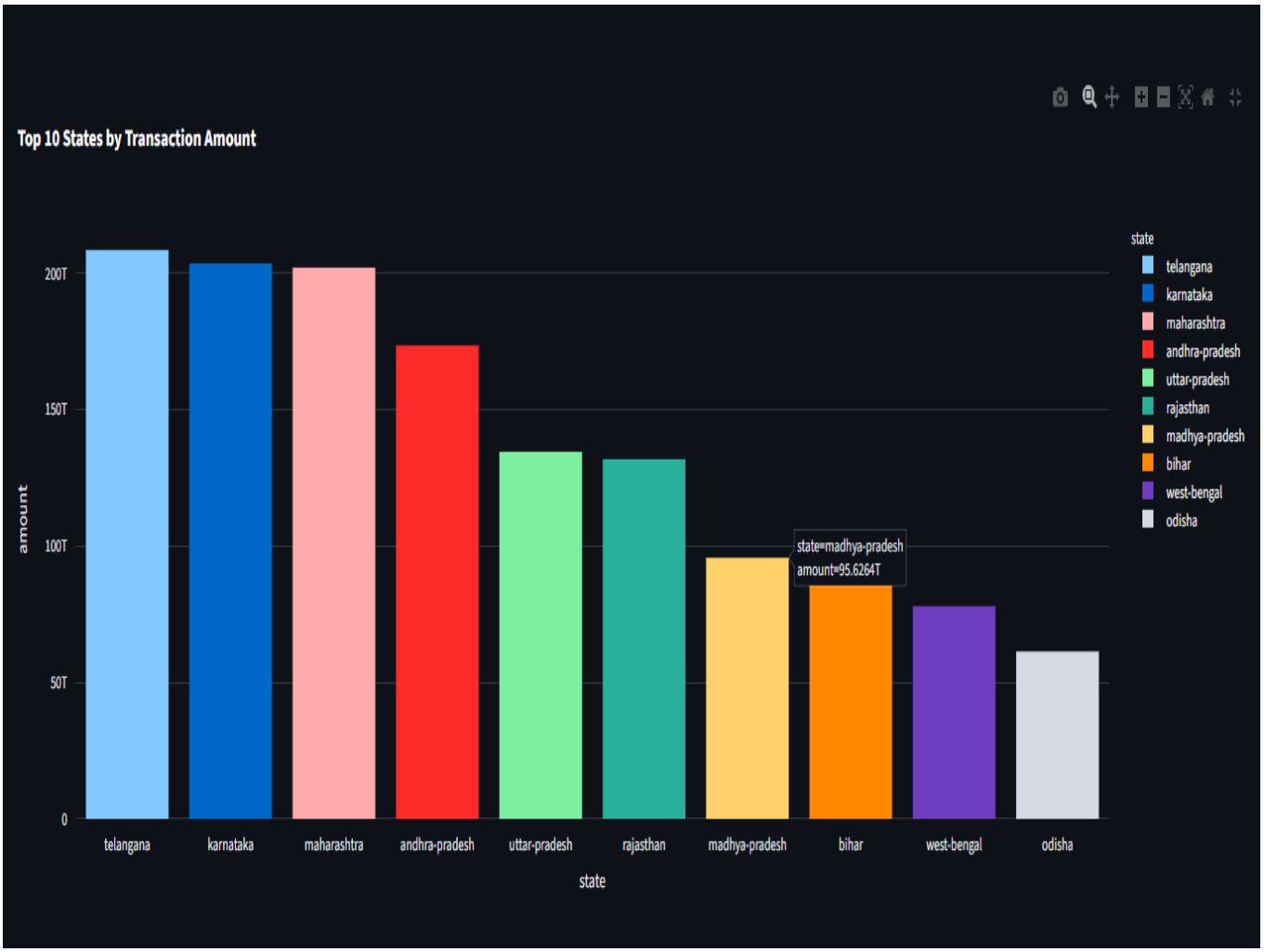
Local URL: http://localhost:8502

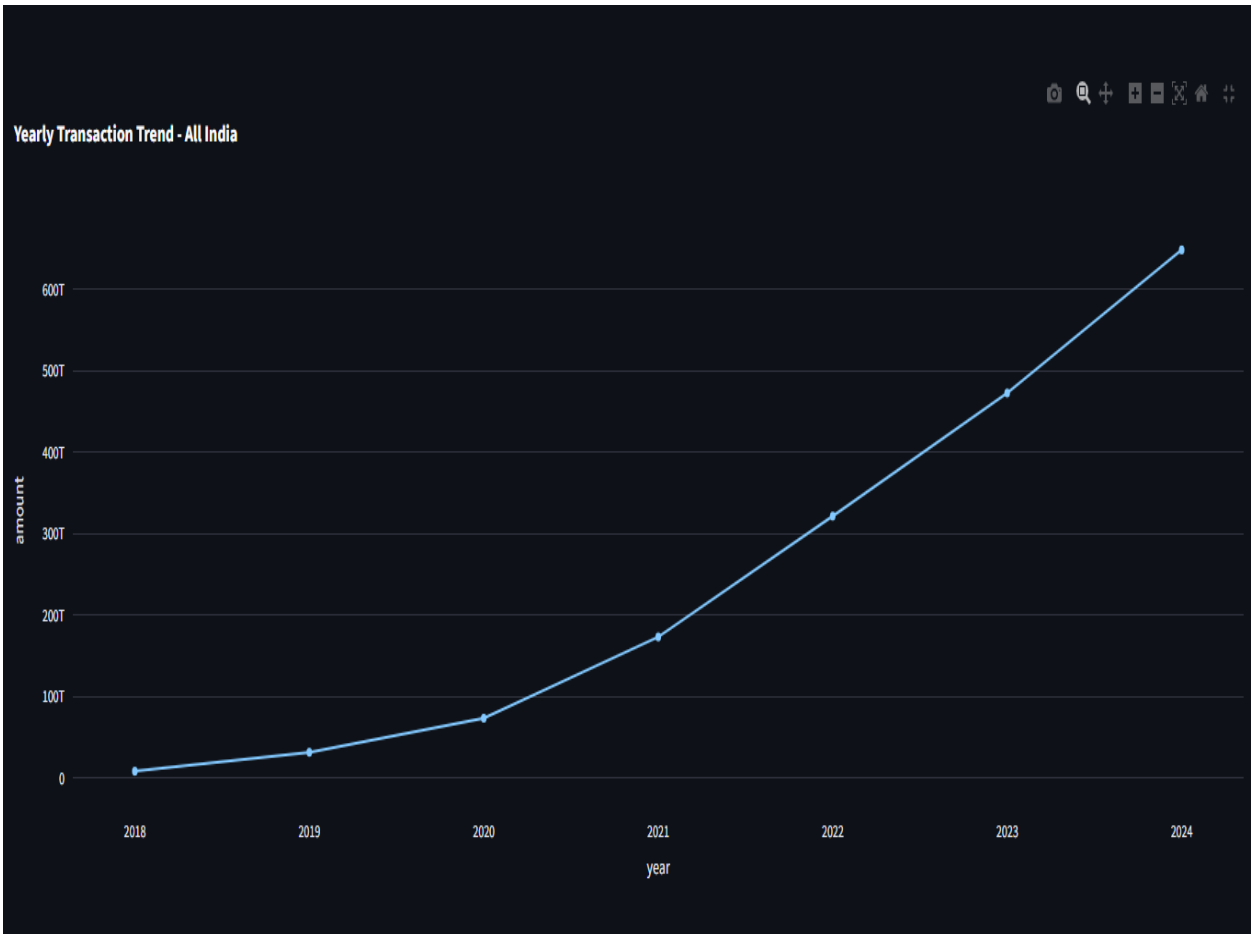
Network URL: http://192.168.1.11:8502

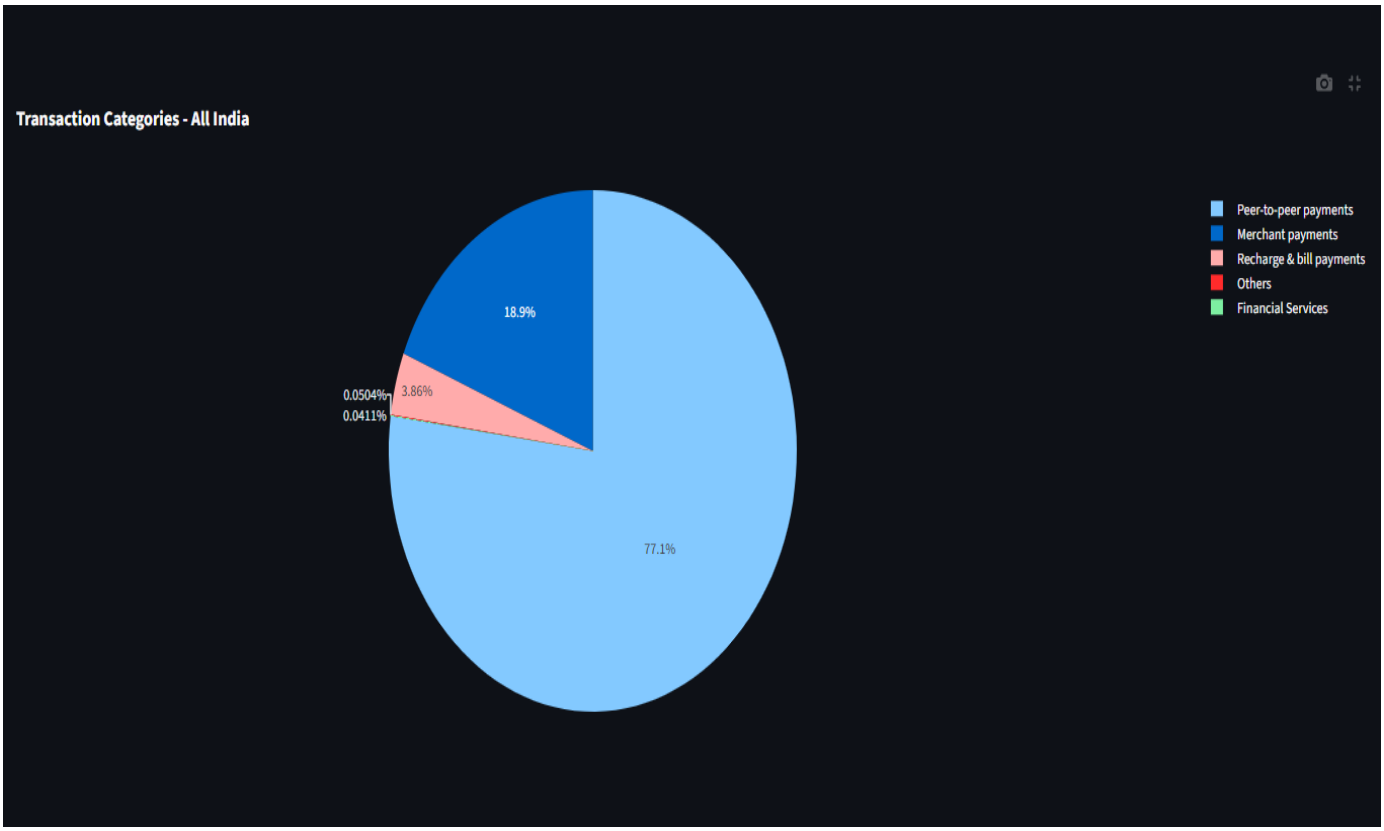
d:\New folder\app.py:18: UserWarning: pandas only supports SQLAlchemy connectable (engine/connection) or database string URI or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider using SQLAlchemy.

df = pd.read_sql(query, conn)

Link og dashbord: PhonePe Transaction Dashboard


Top 10 States by Transaction Amount
amount
200T
150T
100T
50T
0
state=madhya-pradesh
amount=95.6264T
telangana
karnataka
maharashtra
andhra-pradesh
uttar-pradesh
rajasthan
madhya-pradesh
bihar
west-bengal
odisha
state
state
telangana
karnataka
maharashtra
andhra-pradesh
uttar-pradesh
rajasthan
madhya-pradesh
bihar
west-bengal
odisha


Yearly Transaction Trend - All India
600T
500T
400T
300T
200T
100T
0
amount
2018
2019
2020
2021
2022
2023
2024
year


Transaction Categories - All India
Peer-to-peer payments
Merchant payments
Recharge & bill payments
Others
Financial Services
18.9%
3.86%
0.0504%
0.0411%
77.1%

2. Tools & Technologies Used

- Python
- MySQL
- Streamlit
- Pandas, Plotly, etc

3. **Data Source & Structure**

The dataset used in this project has been sourced from the **PhonePe Pulse GitHub repository**, which is an official public data release by **PhonePe**, one of India's leading digital payment platforms. The dataset provides comprehensive insights into the digital transaction patterns across India from 2018 onwards.

The data is stored in a **hierarchical JSON format**, organized by categories such as transactions and users. The structure of the dataset is as follows:

```
pulse-master/
│
├── data/
│   ├── aggregated/
│   │   ├── transaction/
│   │   │   ├── country/
│   │   │   │   └── india/
│   │   │   │       └── state/
│   │   │   │           ├── andhra-pradesh/
│   │   │   │           │   ├── 2018/
│   │   │   │           │   │   ├── 1.json
│   │   │   │           │   │   ├── 2.json
│   │   │   │           │   │   └── ...
│   │   │   │           ├── maharashtra/
│   │   │   │           └── ...
│   │   │   └── ...
```

Each JSON file contains transaction-related data for a specific **state**, **year**, and **quarter**, with details such as:

- **Name** → Type of transaction (e.g., Recharge & Bill Payments, Peer-to-Peer Payments)
- **Count** → Total number of transactions
- **Amount** → Total value of transactions (in INR)

These JSON files were read using Python, and the extracted data was inserted into a **MySQL database** for structured storage and further visualization in **Streamlit**.

4. Methodology / Steps Executed

The project followed a step-by-step process to analyze and visualize the PhonePe Pulse dataset. The major steps executed are as follows:

Step 1: Data Collection

- The dataset was downloaded from the official **PhonePe Pulse GitHub repository**.
- The data was available in **JSON format**, containing state-wise, year-wise, and quarter-wise transaction details.

Step 2: Data Extraction using Python

- Python was used to **read and parse JSON files** using the `json` and `os` libraries.
- The script extracted relevant fields such as `state`, `year`, `name`, `count`, and `amount` from each JSON file.

Step 3: Database Creation and Storage

- A **MySQL database (`phonepe_db`)** was created to store the extracted data.
- Using the `mysql.connector` library, the cleaned data was inserted into structured tables (`aggregated_transaction`).
- Each record represents transaction details for a specific state and year.

Step 4: Data Processing and Cleaning

- Data was checked for missing values and inconsistencies before visualization.
- The dataset was aggregated to get total transaction counts and amounts by state and year.

Step 5: Data Visualization using Streamlit

- A **Streamlit dashboard** was developed to visualize insights interactively.
- The app displays:
    - State-wise and year-wise transaction amounts
    - Top-performing states in digital payments
    - Growth trends over time

Step 6: Analysis and Interpretation

- The generated graphs and charts were analyzed to understand digital transaction growth across Indian states.
- Insights were drawn regarding which regions and categories showed higher transaction activity.

5. **Results & Discussion**

The PhonePe Pulse Data Analysis project successfully visualizes digital payment trends across India using real-world transaction data. The interactive dashboard built with **Streamlit** provides clear and dynamic insights into various transaction categories, states, and time periods.

1. Visualization Results

The Streamlit dashboard displays multiple interactive charts and graphs:

- **State-wise Transaction Overview:**
  Shows total transaction count and amount for each Indian state.
  High-performing states such as **Maharashtra**, **Karnataka**, and **Tamil Nadu** show the highest transaction volumes.
- **Year-wise Growth Analysis:**
  Line and bar charts illustrate the steady growth of digital payments from **2018 to 2024**, reflecting the rapid adoption of UPI and mobile payments.
- **Transaction Type Analysis:**
  Different categories such as **Recharge & Bill Payments**, **Peer-to-Peer Payments**, and **Merchant Transactions** were visualized, revealing that Peer-to-Peer transfers account for the majority of the transaction value.
- **Comparative Insights:**
  The dashboard allows users to compare multiple states and years, helping identify regional variations and digital adoption patterns.

2. Key Observations

- Digital transactions have shown **consistent year-on-year growth** across India.
- **Urban and tech-driven states** such as Maharashtra and Karnataka dominate the digital transaction ecosystem.
- Categories like **Bill Payments** and **Recharge** form the foundation of early adoption, while **Merchant Payments** are rapidly increasing in recent years.
- The visualization highlights how **PhonePe has contributed significantly** to India's digital payment revolution.

6. Conclusion

The *PhonePe Pulse Data Analysis Dashboard* project successfully demonstrates how real-world financial data can be processed, analyzed, and visualized using modern data science tools. By integrating **Python**, **MySQL**, and **Streamlit**, this project provides an end-to-end pipeline — from data extraction and storage to interactive visualization.

Through the analysis, it is evident that **digital payments in India have experienced exponential growth** between 2018 and 2024. States such as **Maharashtra, Karnataka, and Tamil Nadu** show the highest transaction volumes, indicating rapid adoption of digital payment systems in urban and developed regions.

This project not only highlights the potential of open fintech data like **PhonePe Pulse**, but also helps in understanding user behavior and payment trends across India. The interactive dashboard enables data-driven insights, which can assist policymakers, businesses, and researchers in decision-making.

Overall, this project serves as a practical example of applying **data analytics and visualization techniques** to a real industry dataset, bridging the gap between theoretical learning and real-world applications.

Future Work (Optional)

While the current project successfully analyzes and visualizes PhonePe's digital transaction data, there is scope for several enhancements in the future:

1. **District-Level and City-Level Insights:**
   Future versions of the dashboard can include deeper analysis by districts or cities for more granular insights into regional transaction behavior.
2. **Real-Time Data Integration:**
   Connecting the dashboard to real-time APIs could allow continuous updates of transaction trends as new data becomes available.
3. **Advanced Analytics & Machine Learning:**
   Predictive models can be developed to forecast future transaction growth and identify emerging digital payment hubs across India.
4. **User Demographic Analysis:**
   Integration of user demographics such as age, gender, and occupation could provide more personalized insights into user behavior.
5. **Enhanced Visualization Features:**
   Adding advanced filters, heatmaps, and interactive geospatial charts would improve user experience and analytical depth.

8. References

1. **PhonePe Pulse – Official Data Repository**
   https://github.com/PhonePe/pulse
2. **PhonePe Official Website**
   https://www.phonepe.com/pulse/
3. **Python Official Documentation**
   https://docs.python.org/3/
4. **Streamlit Documentation**
   https://docs.streamlit.io/
5. **MySQL Documentation**
   https://dev.mysql.com/doc/
6. **Plotly Visualization Library**
   https://plotly.com/python/