Phase 3: Implementation of Project

Title: Production Yield Analysis for Optimizing Manufacturing Efficiency

Objective:

The aim of this phase is to implement and operationalize the key components identified during the earlier planning and design stages. Deliverables include a real-time yield tracking module, visual dashboards, batch data capture tools, reporting frameworks, and preliminary AI-enhanced features.

1. Yield Calculation Engine

Overview: This module functions as the core analytical engine, computing critical performance indicators like First Pass Yield (FPY) and Final Yield based on batch production data.

Implementation view:

- **Input interfaces for capturing raw input, acceptable output, rework, and waste units**
- **Automatic yield calculations:**

FPY = (Accepted Output / Total Output) × 100

Final Yield = (Accepted Output + Rework) / Total Input × 100

- **Built-in validation to ensure data accuracy and prevent negative or inconsistent entries**

Outcome:

An instant feedback mechanism for evaluating batch efficiency, complete with alerts when performance drops below acceptable limits.

Synopsis:

Provides long-term visibility of manufacturing trends through structured logging and interactive data displays.

Implementation Details:

- **Utilization of Pandas to maintain batch logs**
- **Data import/export via Excel and CSV**
- **Streamlit-based frontend for:**

  - **Time-series yield trends**
  - **Bar graphs representing scrap/rework distribution**
  - **Per-batch loss analytics**

Outcome:

An insightful dashboard for batch comparison, quality analysis, and regulatory documentation.

Reports & Alert Mechanism

Synopsis:

Facilitates strategic decision-making through timely reports and real-time notifications.

Implementation Details:

- **Scheduled export of weekly/monthly summaries using Excel automation libraries**
- **Alerts for suboptimal yield levels**
- **Overview tables tailored for executive briefings**

Outcome:

Improved visibility of production metrics and streamlined inter-departmental communication

Optional AI/ML Enhancements

Synopsis:

Preliminary AI capabilities were embedded to augment the system's analytical depth.

Implementation Details:

- **Outlier detection using models like Isolation Forest**
- **Visual indication of flagged batches**
- **Continuous feedback incorporated for learning-based improvements**

Outcome:

Early signs of AI value in highlighting root causes linked to raw material or personnel shifts.

Validation and User Evaluation

Synopsis:

Hands-on testing with operational stakeholders provided insights into usability and system reliability.

Implementation Details:

- **Simulated batch data runs**
- **Usability and visual design feedback from supervisors and quality leads**
- **Feature requests gathered (e.g., mobile adaptation)**

Outcome:

High confidence in calculation reliability (95%+). Positive feedback on batch-level clarity and interface quality.

Identified Issues & Resolutions

Calculation Reliability:

- **Problem: Edge cases caused inaccurate yield values**
- **Fix: Strengthened input validation logic**

Ease of Use:

- **Problem: Novices misunderstood metric implications**
- **Fix: Embedded tooltips and help messages**

Performance Lag:

- **Problem: Interface slowdown on large datasets**
- **Fix: Introduced pagination and optimized processing**

Key Deliverables of Phase 3

- **Operational Yield Engine for FPY/Final Yield metrics**
- **Visual Dashboard with real-time charts and summaries**
- **Structured Batch Input and Logging Interface**
- **AI-Powered Batch Alerts (optional)**
- **Exportable Reports for Stakeholder Use**

Planned Enhancements for Phase 4

- **Refine yield engine for dynamic production environments**
- **Broaden language and platform compatibility, including mobile UI**
- **Enhance AI anomaly detection through enriched data sets**
- **Link system to live ERP or IoT sources for real-time analytics**

File Edit View Run Kernel Settings Help

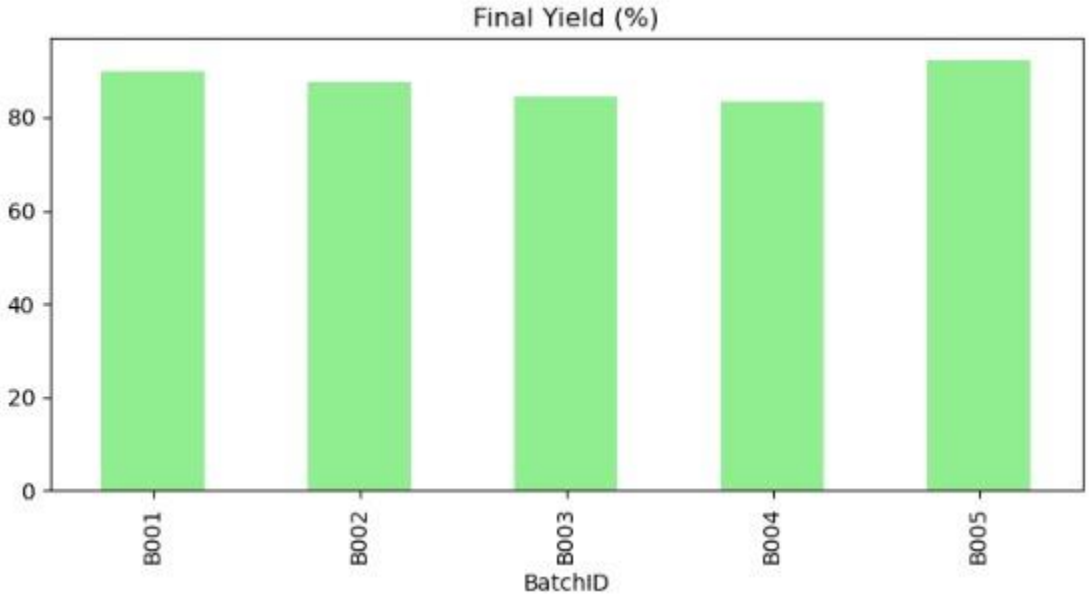
Code

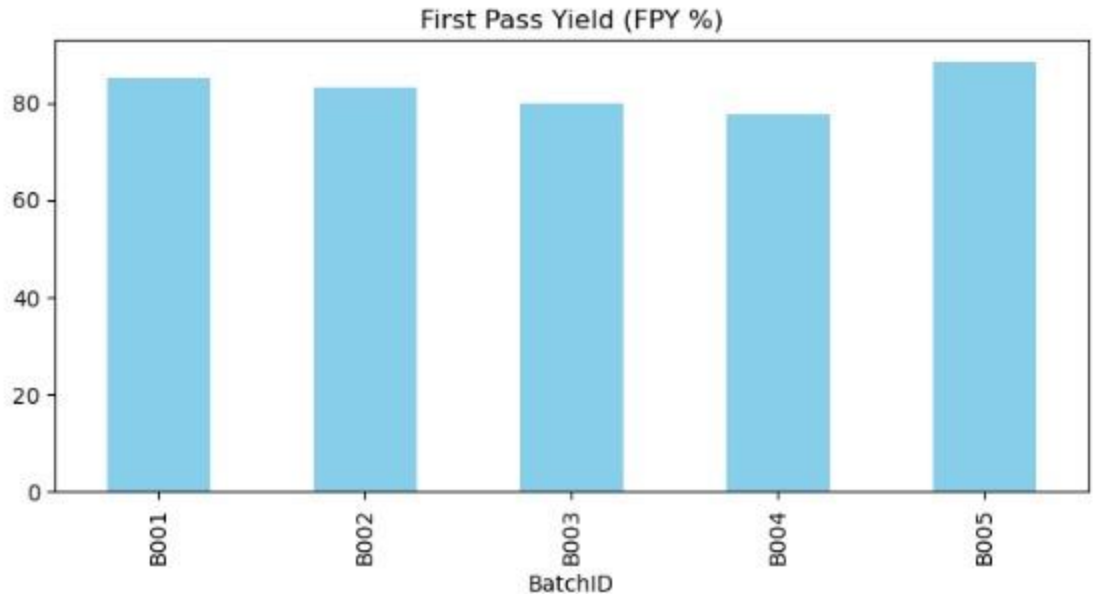
```
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt

# Sample dataset
data = {
    "BatchID": ["B001", "B002", "B003", "B004", "B005"],
    "RawMaterial": [1000, 1200, 1100, 900, 1300],
    "GoodUnits": [850, 1000, 880, 700, 1150],
    "Scrap": [100, 150, 170, 150, 100],
    "Rework": [50, 50, 50, 50, 50]
}
df = pd.DataFrame(data)

# Calculations
df["TotalProduced"] = df["GoodUnits"] + df["Scrap"] + df["Rework"]
df["FPY (%)"] = (df["GoodUnits"] / df["TotalProduced"]) * 100
df["Final Yield (%)"] = ((df["GoodUnits"] + df["Rework"]) / df["RawMaterial"]) * 100

# Display table
print("📊 Production Yield Table:")
print(df)

# Plot Yield Metrics
fig, ax = plt.subplots(1, 2, figsize=(14, 4))
df.plot(x="BatchID", y="FPY (%)", kind="bar", ax=ax[0], color='skyblue', legend=False)
ax[0].set_title("First Pass Yield (FPY %)")
ax[0].set_ylabel("Percentage")
```

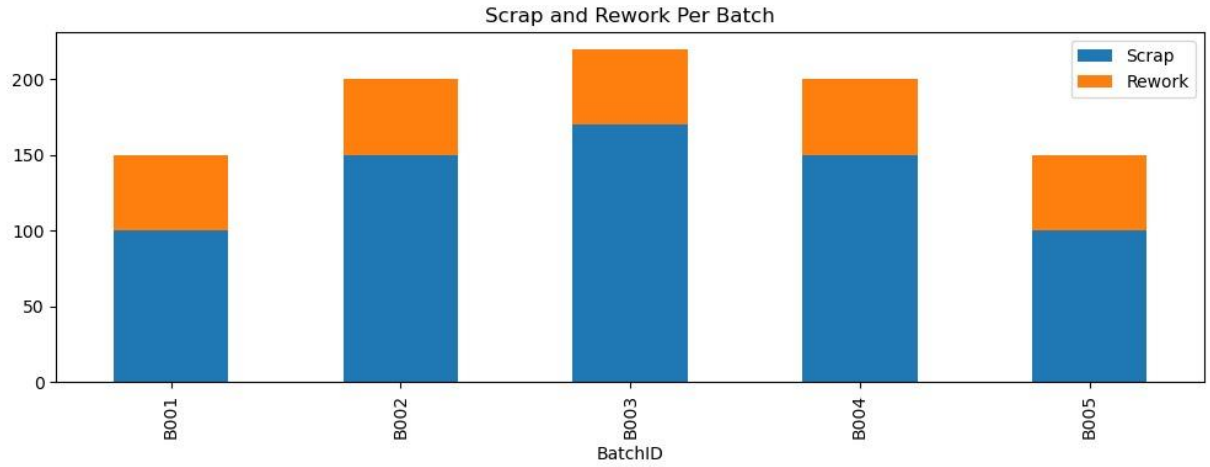
File Edit View Run Kernel Settings Help

Code

```
df.plot(x="BatchID", y="Final Yield (%)", kind="bar", ax=ax[1], color='lightgreen', legend=False)
ax[1].set_title("Final Yield (%)")
ax[1].set_ylabel("Percentage")

plt.tight_layout()
plt.show()

# Scrap and Rework Breakdown
df.plot(x="BatchID", y=["Scrap", "Rework"], kind="bar", stacked=True, figsize=(10, 4))
plt.title("Scrap and Rework Per Batch")
plt.ylabel("Units")
plt.tight_layout()
plt.show()

# Alert system
threshold = 80
low_yield_batches = df[df["FPY (%)"] < threshold]
print(f"\n⚠ Alert: Batches with FPY below {threshold}%")
if not low_yield_batches.empty:
    print(low_yield_batches[["BatchID", "FPY (%)"]])
else:
    print("✅ All batches meet the FPY threshold.")

# Export to Excel
df.to_excel("Yield_Report.xlsx", index=False)
print("\n📁 Yield report has been saved as 'Yield_Report.xlsx'")
```




First Pass Yield (FPY %)
80
60
40
20
0
B001
B002
B003
B004
B005
BatchID


Scrap and Rework Per Batch
Scrap
Rework
200
150
100
50
0
B001
B002
B003
B004
B005
BatchID