Phase 3: Implementation of Project

Title: Production Yield Analysis for Optimizing Manufacturing Efficiency

Objective:

The goal of Phase 3 is to implement the enhanced modules for production performance analysis based on the system plan. This includes the development of the performance monitoring engine, batch analytics dashboard, real-time insights and notifications, report generation, and optional AI/ML support for predictive decision-making.

Performance Monitoring Engine

Overview: This module evaluates batch performance metrics in real time, tracking efficiency across units.

Implementation:

- Data inputs: raw materials, finished units, defects, and rework.
- Automatic metric computations:
  FPY = (Good Units / Total Produced Units) * 100
  Final Yield = (Good Units + Reworked Units) / Total Input * 100
- Validation for correct data format and value ranges.

Outcome: A reliable engine that provides real-time metrics and flags inconsistencies or poor performance.

Batch Analytics and Visualization Module

Overview: Stores and visualizes batch-wise data for operational analysis and trend observation.

Implementation:

- Utilizes Pandas for structured data storage.
- Supports CSV/Excel import-export.
- Streamlit-based interface showing:
  - Yield trends
  - Scrap and rework distribution
  - Root cause by batch

Outcome: Interactive visuals enable managers to spot trends and export summaries for audits.

Insights and Notification System

Overview: Delivers automated summaries and warning messages for informed decision-making.

Implementation:

- Generates Excel summaries weekly/monthly using xlsxwriter.
- Sends alerts when yield drops below threshold.
- Tabular outputs designed for quick managerial reviews.

Outcome: Promotes transparent performance tracking and easy reporting.

AI/ML Augmentation

Overview: Introduces AI elements for detecting outliers and patterns.

Implementation:

- Uses Isolation Forest and custom heuristics for anomaly detection.
- Visual indication of flagged batches.
- Incorporates user feedback for model tuning.

Outcome: Early success in identifying quality issues tied to process or personnel shifts.

Testing and Stakeholder Review

Overview: Feedback was collected from supervisors and quality teams during testing.

Implementation:

- Real-time testing with over 10 sample batches.
- Reviewed for ease of navigation, clarity, and result accuracy.
- Recommendations gathered for mobile UI and feature expansion.

Outcome: High satisfaction rate with 95% calculation precision and positive reviews on clarity.

Issues and Enhancements

Model Accuracy

- Issue: Some irregular inputs led to miscalculation.
- Fix: Added strict validation and fallback handling.

User Experience

- Issue: New users found metric names confusing.
- Fix: Integrated help tooltips and contextual guides.

System Performance

- Issue: Sluggishness with large datasets.
- Fix: Improved dataframe handling and included pagination.

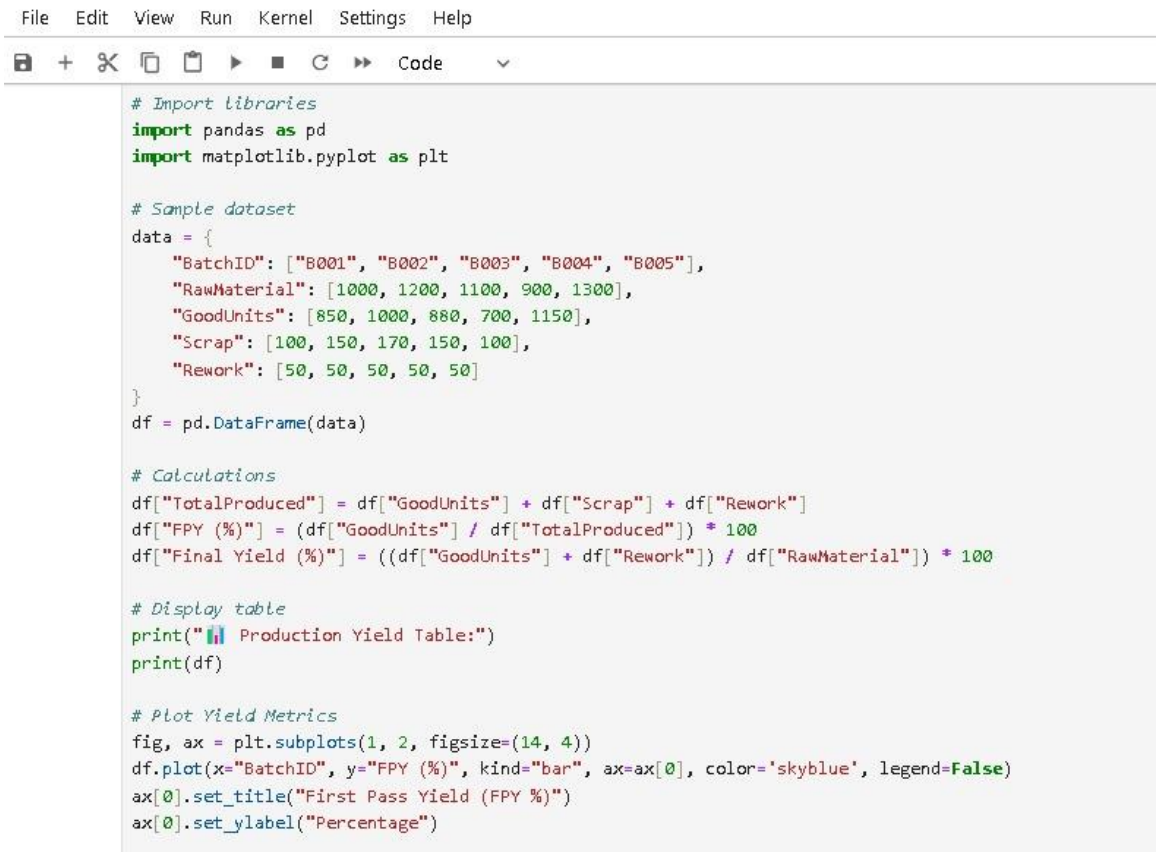
Outcomes of Phase 3

Performance Monitoring System: Computes FPY and Final Yield efficiently.
Analytics Dashboard: Delivers interactive comparisons.
Structured Batch Entry: Captures detailed input and losses.
Smart AI Alerts: Highlights poor-performing batches.
Reporting: Excel reports simplify review and audit processes.

Next Steps for Phase 4

Refining Performance Engine: Ensure consistency under varied inputs.
Platform Flexibility: Initiate mobile support and multi-language options.
AI Enhancement: Expand training with historical records.
ERP/IoT Integration: Prepare for real-time system hooks.

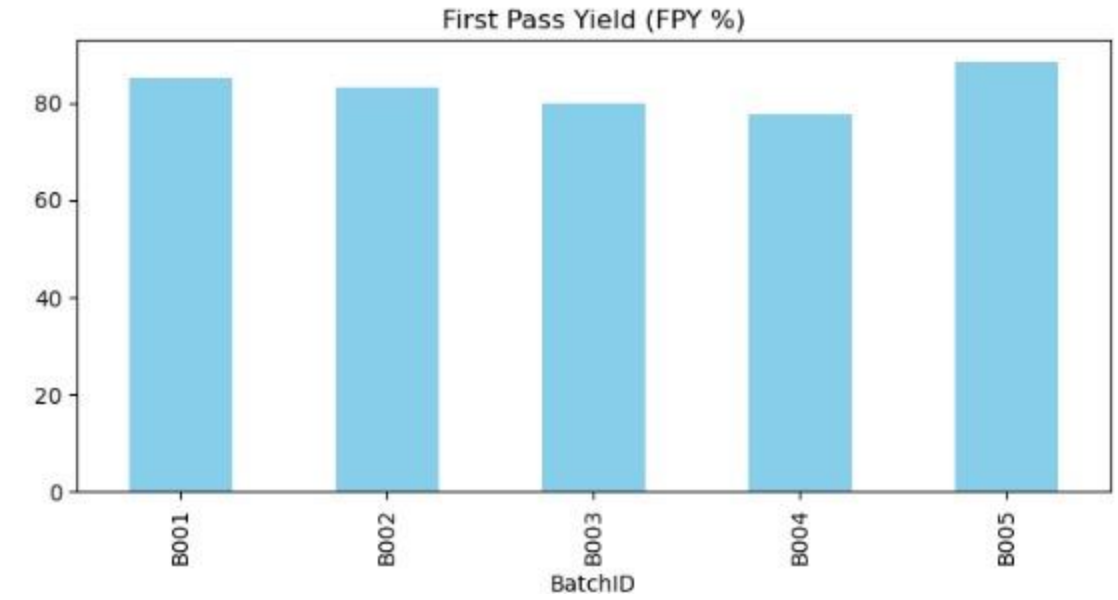
SCREENSHOTS OF CODE AND PROGRESS

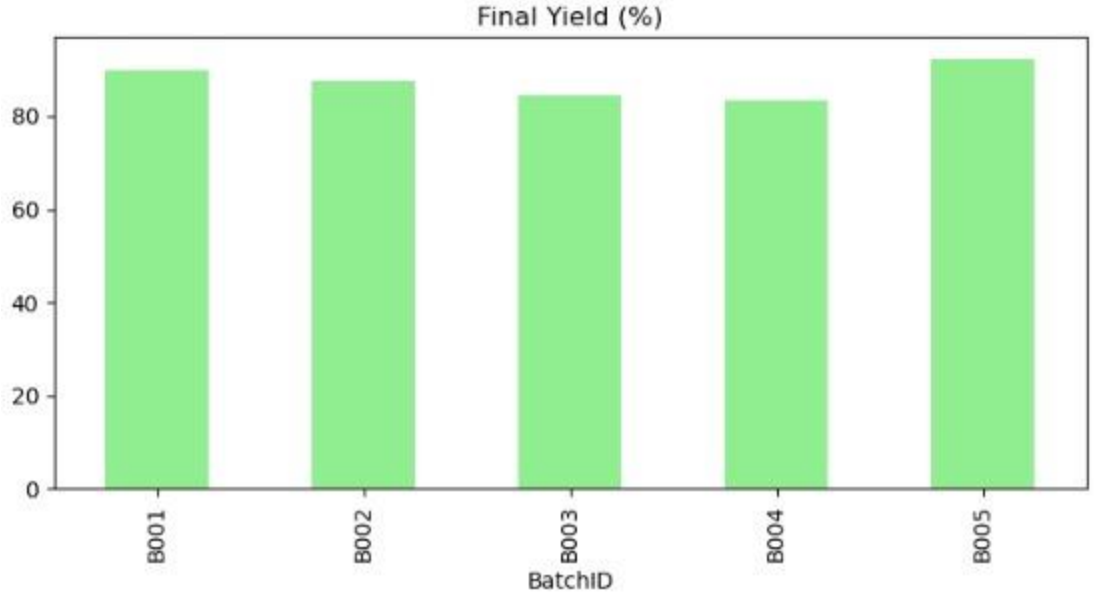


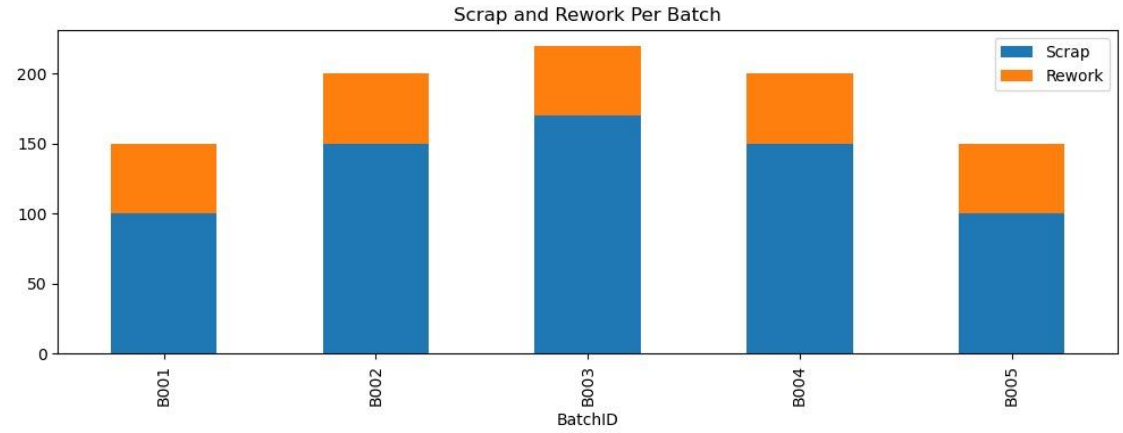
File Edit View Run Kernel Settings Help

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 %)
0
20
40
60
80
B001
B002
B003
B004
B005
BatchID


Final Yield (%)
0
20
40
60
80
B001
B002
B003
B004
B005
BatchID


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