Milestone 3 Report

1. Introduction

Milestone 3 focused on deploying the predictive maintenance model as an interactive web application, integrating advanced visualizations, and ensuring smooth end-to-end inference. The objective was to convert the trained RUL prediction model into a user-friendly, explainable, and operational tool accessible through a web dashboard. This milestone marks the transition from model development to practical usability.

---

2. Work Completed in Milestone 3

2.1 Model Loading and Inference Pipeline Integration

The trained BiLSTM RUL prediction model and associated scalers were successfully integrated into a Flask backend. Key tasks completed include:

- Packaging the model into `.h5` format
- Loading feature scaler files for consistent preprocessing
- Creating an inference function to process new engine data files
- Ensuring correct feature ordering and padding for model input
- Handling fallback pipelines when scalers are unavailable

The full inference pipeline is now automated, from file upload to final RUL prediction.

2.2 Web Application Development (Flask + HTML + CSS)

A complete web dashboard was designed and implemented to allow users to upload turbofan engine data and obtain predictions.

Functional components built:

- File upload interface
- Backend processing and validation of CSV files
- Display of predicted RUL
- Engine status classification (Healthy, Moderate Risk, Critical)
- Pipeline summary and model metadata

The interface was designed to be simple, intuitive, and responsive.

2.3 Visual Analytics Integration

To enable engineers to interpret predictions, multiple interactive visualizations were incorporated into the dashboard using Plotly.

Implemented visual components:

- Sensor Trends (first 6 sensors)
- Cycle-wise Degradation (top degrading sensors)
- RUL Gauge (color-coded)
- Health Metric Indicator
- Detailed heatmap of sensor correlations

These visualization components support exploratory analysis and help users understand both engine condition and model outputs.

2.4 Heatmap Generation via Matplotlib

A server-side correlation heatmap was implemented using Matplotlib and Seaborn.
Key work included:

- Computing correlation matrices from preprocessed sensor data

- Rendering the heatmap on the server
- Encoding the heatmap as an image for display in the dashboard

This offers insight into relationships between sensors and helps identify redundant or highly dependent measurements.

2.5 Bug Fixing and Robust JSON Serialization

Several issues related to:

- Numpy array serialization
- Plotly JSON encoding
- Data type mismatches
- Sequence creation errors
- Feature misalignment

were identified and resolved.
Fixes included:

- Converting all arrays to Python lists
- Ensuring Plotly receives valid JSON objects
- Adding checks for missing or inconsistent sensor columns
- Guaranteeing cycle indexing is valid for all inputs

This ensures stable operation and consistent visualization across all uploads.

2.6 PDF Report Export (Planned but Partially Implemented)

Initial work was completed on generating downloadable PDF reports containing:

- Predicted RUL
- Health classification
- Summary statistics
- Embedded visualizations

The core functionality has been developed, and final formatting will be completed in the next milestone.

2.7 End-to-End Testing

The system was tested across:

- Multiple FD001–FD004 engine files
- Different sequence lengths
- Fallback pipelines (scaler vs raw)
- Invalid or malformed input files

All major functional paths are working, and predictions are reliably generated.

---

3. Summary of Milestone 3 Outcomes

The key outcomes of Milestone 3 include:

- Deployment of the model through a Flask-based web dashboard
- A complete preprocessing + inference + visualization pipeline
- Interactive and interpretable visual analytics
- Robust handling of different file formats and datasets
- Strong user experience with live plots and indicators

- Partial implementation of PDF reporting

This milestone transformed the machine learning model into a practical, usable predictive maintenance application.

---

4. Work Planned for Milestone 4

Milestone 4 will focus on finalizing deployment usability, enhancing explainability, and preparing the project for submission or demonstration.

4.1 Full PDF Reporting Module

- Generate complete RUL reports
- Include sensor trends, heatmap, and degradation curves
- Add maintenance recommendations
- Provide downloadable summaries for each engine

4.2 Dashboard Enhancements

- Add engine-wise comparison tools
- Allow multi-file uploads
- Add advanced analytics such as:
    - Degradation rate estimation
    - Predictive thresholds
    - Severity maps

4.3 Explainability and Model Transparency

- Incorporate SHAP or LIME to explain feature importance

- Visualize sensor contributions to each RUL prediction
- Provide interpretability reports

4.4 Model Optimization (If Required)

- Additional tuning of BiLSTM layers
- Experimentation with hybrid CNN-LSTM models
- Performance benchmarking

4.5 Final Project Documentation

- Complete final report
- Prepare PPT slides
- Create demonstration script
- Package codebase for submission

---

5. Conclusion

Milestone 3 successfully delivered the deployment phase of the project. The predictive model now operates seamlessly through a web dashboard with comprehensive visual analytics, enabling clear interpretation of engine health and RUL predictions. The foundation laid in this milestone sets the stage for the final deliverables in Milestone 4, including reporting, explainability, and enhanced visual intelligence.