Here is the \*\*clean bullet-point version formatted for easy inclusion in a single PPT slide\*\*:

---

### 🌟 \*\*Additional Information\*\*

\* 📊 \*\*Dataset Used:\*\*

Supplier Quality & Performance Dataset

▶️ [https://github.com/Rolakamin/Supplier-Quality-and-Performance](https://github.com/Rolakamin/Supplier-Quality-and-Performance)

\* ⚡ \*\*Cold-Start Problem:\*\*

New suppliers have no historical performance data → Reliability predicted using metadata similarity.

\* 🔍 \*\*Evaluation Metrics:\*\*

• Accuracy

• Precision

• Recall

• F1-score

• Confusion Matrix

\* 🔧 \*\*Tools & Technologies:\*\*

• Python (Pandas, NumPy, scikit-learn, matplotlib, seaborn)

• Joblib (model saving)

• Streamlit / Flask (for deployment)

\* 🚀 \*\*Future Work Ideas:\*\*

• Real-time data integration for continuous updates.

• Graph Neural Networks for advanced cold-start handling.

• Adding verified certification data as a feature.

---

👉 Would you like me to create a ready-to-use PowerPoint slide (.pptx) file with this content styled neatly?

Here is a simple and clear \*\*3-module structure for your project\*\*, covering from scratch to deployment:

---

## ✅ \*\*Module 1 – Data Preparation & Exploration\*\*

\* Collect the Supplier Quality & Performance Dataset.

\* Clean the dataset by handling missing values.

\* Encode categorical features (Material Type, Category, Defect Type, Plant Location).

\* Normalize numerical features (Defect Quantity, Downtime, Defect Rate).

\* Perform Exploratory Data Analysis (EDA):

• Visualize defect rate distribution.

• Analyze feature correlations.

• Understand data patterns.

---

## ✅ \*\*Module 2 – Model Development\*\*

\* Build the \*\*supervised machine learning model\*\* to predict supplier reliability using historical data.

\* Train and evaluate the model on labeled data (Reliable / Unreliable).

\* Implement the \*\*cold-start prediction logic\*\*:

• Use metadata features to find similar existing suppliers.

• Apply similarity-based prediction (e.g., KNN) for new suppliers with no history.

\* Validate model performance using accuracy, precision, recall, and F1-score.

\* Save trained model and preprocessing tools (scaler, label encoders).

---

## ✅ \*\*Module 3 – Deployment & Reporting\*\*

\* Develop a simple interface (e.g., Streamlit / Flask web app) where users can:

• Upload new supplier metadata.

• Receive a reliability prediction in real time.

\* Generate reports and visualizations:

• Feature importance.

• Reliability distribution of existing and new suppliers.

\* Provide insights and recommendations for procurement decisions.

\* Plan for future improvements (real-time updates, graph-based methods).

---

👉 Would you like me to generate a \*\*flowchart diagram\*\* visualizing these 3 modules for your presentation slide?

The \*\*perfect number of modules\*\* to split your project depends on clarity, logical separation of concerns, and manageability.

👉 For your project, a clean and well-balanced approach is to split it into \*\*3 main modules\*\*:

---

### ✅ 1️⃣ – \*\*Data Preparation & Exploration\*\*

• Collect and clean the dataset

• Handle missing values and encode features

• Perform Exploratory Data Analysis (EDA)

• Feature engineering (e.g., Defect Rate)

---

### ✅ 2️⃣ – \*\*Model Development & Cold-Start Handling\*\*

• Train supervised machine learning model for existing supplier evaluation

• Implement cold-start prediction logic (similarity-based using metadata)

• Evaluate model performance (Accuracy, Precision, Recall, F1-score)

• Save trained models and preprocessing tools

---

### ✅ 3️⃣ – \*\*Deployment & Reporting\*\*

• Build a simple interface (Streamlit / Flask) for real-time prediction

• Generate visualization reports (feature importance, reliability distribution)

• Provide actionable insights and recommendations for decision makers

• Plan for future scalability (e.g., Graph Neural Networks, real-time updates)

---

### ⚡ Why 3 Modules Are Perfect

✔️ Covers the entire pipeline from scratch to deployment

✔️ Simple enough to manage without overwhelming complexity

✔️ Logically separates data, modeling, and deployment responsibilities

✔️ Flexible for future extensions without redesigning the structure

---

👉 Would you like me to generate a visual flowchart of this 3-module structure for your presentation?