What D2 Contains

D2 is the **working AI prototype** — all the **Python code, data, and a pretrained RandomForest model** that demonstrates how Acme SmartPay AP could automate **invoice–PO/GRN matching**.

It shows:

1. How data flows in
2. How we build features from invoices/POs
3. How the model predicts **matches vs mismatches**
4. How results are evaluated

👷 **Pipeline Walkthrough (D2)**

1. Data Loading (src/data_loader.py)

- Reads three CSVs:
  - **invoices.csv** → line-level invoice details (vendor, date, currency, line totals)
  - **po_grn.csv** → purchase order (PO) + goods receipt (GRN) data
  - **labelled_mismatches.csv** → human-labeled truth set (which invoice matched or mismatched its PO)

👉 This mimics how Acme's ERP/Document pipeline would provide raw financial documents.

---

2. Feature Engineering (src/features.py)

We **join invoices ↔ PO/GRN ↔ mismatches** and compute **features** that matter for finance teams:

- amount_diff → absolute difference between invoice total and PO total
- amount_ratio → relative difference (helps with scaling issues)
- date_diff_days → days between invoice date and PO date (flagging late invoices)
- vendor_match → whether invoice vendor matches PO vendor
- currency_match → currency consistency check
- item_count → number of items in the invoice

👉 These features encode **business rules** that AP teams usually check manually.

---

3. Model Training (src/model.py)

We train a **RandomForestClassifier** to detect mismatches:

- Input (X): the **features above**
- Target (y): the **label** from labelled_mismatches.csv
    - **0 = Correct Match** ✅
    - **1 = Mismatch** ❌ (needs review by AP team)

Why RandomForest?

- It handles **non-linear relationships** (e.g., big amount differences + vendor mismatch = more likely fraud).
- Works well on small-to-medium tabular datasets like invoices.
- More robust than Logistic Regression (which we started with earlier).

---

4. Scripts

- **train_model.py** → trains a model from scratch on provided data and saves it as models/matcher.pkl.
- **evaluate_model.py** → loads the pretrained model and prints accuracy, precision, recall, F1-score, and confusion matrix.

---

5. What the Model Predicts

- For every invoice–PO pair:
    - **Prediction = 0** → invoice matches PO/GRN (safe to auto-process 💸)
    - **Prediction = 1** → mismatch detected (send to AP officer for investigation 🔍)

👉 This saves finance teams **hours of manual reconciliation** by only flagging exceptions.

---

6. Evaluation Metrics

Example (from the pre-trained RandomForest in your repo):

Classification Report (Test Split):

Evaluation Report (Full Set):

	precision	recall	f1-score	support
0	0.7857	0.9167	0.8462	12
1	0.9333	0.8235	0.8750	17
accuracy			0.8621	29
macro avg	0.8595	0.8701	0.8606	29
weighted avg	0.8722	0.8621	0.8631	29

Confusion Matrix:

[[11 1]

[3 14]]