**AI Apa yang Kamu Pake?**

1. **Machine Learning Klasik (Supervised Learning)**
   * Kamu punya dataset (troubleshooting\_dataset.csv) dengan kolom question, answer, intent.
   * Model **Logistic Regression** dilatih untuk mengenali **intent** dari pertanyaan user.
   * Jadi modelny itu klasifikasi: dari teks → intent.
2. **NLP (Natural Language Processing)** dasar
   * Pertanyaan user diproses pakai **TF-IDF Vectorizer** → mengubah teks jadi angka.
   * Dipakai oleh model Logistic Regression buat prediksi intent.
3. **Retrieval-based Chatbot**
   * Chatbot tidak “menciptakan jawaban baru”.
   * Chatbot **mengambil jawaban yang sudah ada di dataset** berdasarkan intent yang diprediksi.
   * Kalau confidence rendah, chatbot pakai **semantic similarity (cosine similarity)** untuk cari pertanyaan paling mirip.

**Retrieval-based Chatbot (kayak punyamu sekarang)**

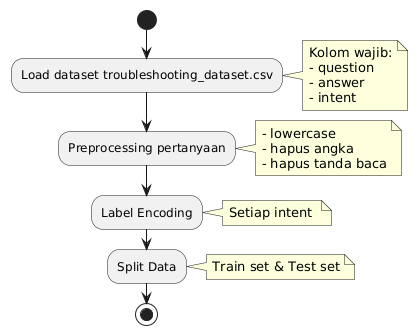
* **Data:** ambil dari dataset internal (question, answer, intent).
* **Teknologi:** ML klasik (Logistic Regression, TF-IDF, SVM, Naive Bayes, dsb).
* **Output:** selalu pilih jawaban dari daftar yang sudah ada → **tidak bikin jawaban baru**.
* **API?** ❌ Tidak perlu API eksternal, cukup dataset & model di laptop/server.

**🔹 Generative AI Chatbot (contoh: ChatGPT, Gemini, Claude, LLaMA)**

* **Data:** model sudah dilatih di miliaran teks.
* **Teknologi:** Large Language Model (LLM) → deep learning dengan miliaran parameter.
* **Output:** bisa bikin jawaban baru yang fleksibel, bahkan kalau pertanyaan belum ada di dataset.
* **API?**
  + ✅ Bisa pakai API (misalnya OpenAI API, Gemini API) → gampang dipakai tapi harus internet & bayar.
  + ❌ Bisa juga **jalan lokal** tanpa API, asal kamu install LLM di laptop/server (contoh: **LLaMA 2, Mistral, Falcon** pakai library seperti transformers atau llama.cpp).  
    Tapi butuh GPU/RAM besar.

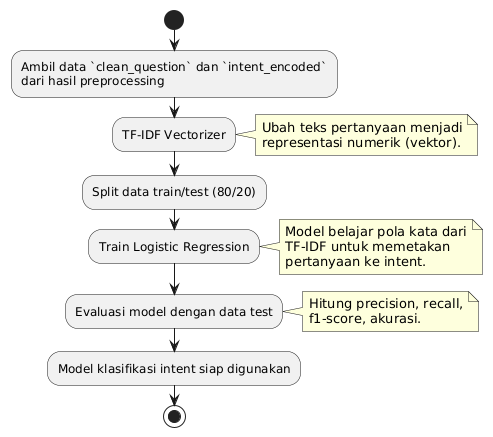
**Alur**

* + 1. **Persiapan Data**

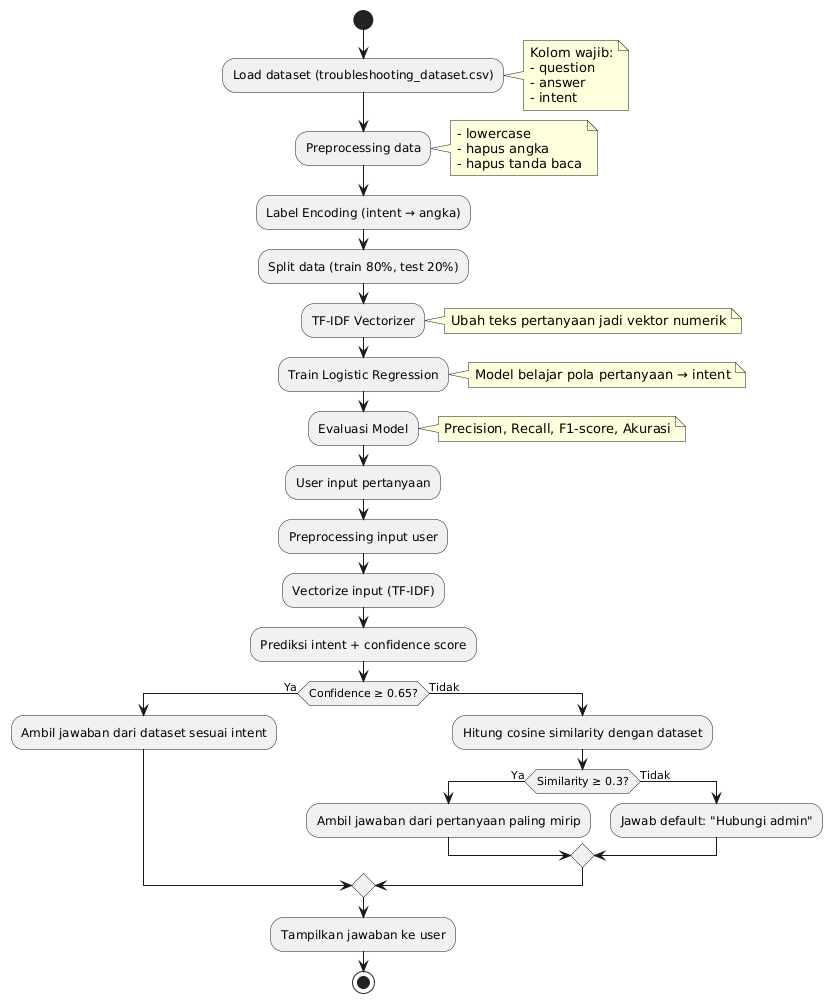


**Persiapan Data**

* **Input:** file troubleshooting\_dataset.csv (kolom question, answer, intent).
* **Proses:**
  1. **Preprocessing**
     + lowercase → huruf kecil semua
     + hapus angka
     + hapus tanda baca
     + simpan ke kolom clean\_question, clean\_answer
  2. **Label Encoding**
     + intent (teks) → angka (intent\_encoded)
  3. **Split data**
     + Train set = 80%
     + Test set = 20%
* **Output:** dataset siap latih (train, test) + troubleshooting\_preprocessed.csv.
  + 1. **Pelatihan Model**



* **Input:** clean\_question (TF-IDF vector) dan intent\_encoded.
* **Proses:**
  + Gunakan **TF-IDF Vectorizer** → representasi teks dalam bentuk numerik.
  + Latih **Logistic Regression** untuk memprediksi intent.
* **Output:** model klasifikasi intent.
  + 1. Evaluasi



Sistem chatbot troubleshooting ini bekerja melalui dua tahap utama. Pertama, data dari file troubleshooting\_dataset.csv diproses dengan mengubah semua teks menjadi huruf kecil, menghapus angka, serta tanda baca agar data lebih bersih. Setiap intent kemudian diubah menjadi angka melalui proses label encoding, lalu data dibagi menjadi 80% untuk pelatihan dan 20% untuk pengujian. Pertanyaan dilatih menggunakan metode TF-IDF Vectorizer yang mengubah teks menjadi representasi angka, dan model Logistic Regression digunakan untuk mempelajari pola hubungan antara pertanyaan dan intent. Setelah model dilatih, akurasi dan kinerjanya dievaluasi menggunakan metrik seperti precision, recall, dan f1-score.

Pada tahap penggunaan, chatbot menerima input pertanyaan dari pengguna, kemudian melakukan preprocessing dan mengubahnya menjadi vektor TF-IDF. Model memprediksi intent pertanyaan sekaligus memberikan tingkat keyakinan (confidence score). Jika tingkat keyakinan ≥ 0.65, chatbot langsung mengambil jawaban sesuai intent dari dataset. Namun, jika confidence rendah, sistem akan menggunakan cosine similarity untuk mencari pertanyaan paling mirip di dataset. Bila ditemukan kemiripan yang cukup (≥ 0.3), maka jawaban dari pertanyaan tersebut diberikan, tetapi jika tidak ada yang mirip maka chatbot akan menjawab dengan pesan standar untuk menghubungi admin. Dengan alur ini, chatbot tetap mampu memberikan jawaban meskipun model tidak sepenuhnya yakin terhadap prediksi intent.

1. User tanya → teks diproses → jadi **vektor TF-IDF**.
2. Vektor ini masuk ke **Logistic Regression** → keluar prediksi intent + confidence score.
3. **Decision:**
   * Kalau **confidence ≥ 0.65** → langsung ambil jawaban berdasarkan intent.
   * Kalau **confidence < 0.65** → sistem **tetap pakai vektor TF-IDF yang sama**, tapi alurnya pindah ke **cosine similarity**:
     + Bandingin vektor pertanyaan user dengan semua vektor pertanyaan dataset.
     + Cari pertanyaan paling mirip.
     + Kalau similarity ≥ 0.3 → ambil jawabannya.
     + Kalau < 0.3 → balas default "Hubungi admin".