Dokumentasi SDLC Federated Learning – *Subsidi Eligibility Prediction System*

1. Pendahuluan

Sistem *Federated Learning* Penentuan Kelayakan Subsidi dikembangkan untuk membantu pemerintah dalam menentukan siapa yang layak menerima bantuan sosial secara objektif, aman, dan efisien.
Proyek ini melibatkan tiga instansi utama yang masing-masing memiliki sumber data berbeda:

- **DINSOS** → Data ekonomi & kondisi rumah tangga
- **DUKCAPIL** → Data kependudukan, pekerjaan, & status keluarga
- **KEMENKES** → Data kesehatan, gizi, & riwayat penyakit

Pendekatan yang digunakan adalah Federated Learning berbasis TensorFlow Federated (TFF), di mana setiap instansi melatih model lokal tanpa membagikan data mentah. Bobot model dikirim ke server pusat untuk dilakukan Federated Averaging (FedAvg). Hasil akhir berupa model gabungan yang dapat digunakan melalui REST API Flask untuk prediksi real-time.

2. Tahapan SDLC

Fase	Deskripsi
Planning	Menentukan kebutuhan sistem, tujuan proyek, serta batasan privasi antar instansi (DINSOS, DUKCAPIL, KEMENKES).
Analysis	Mengidentifikasi kebutuhan data, variabel penting, serta merancang aturan kelayakan berbasis logika (rule-based).
Design	Mendesain arsitektur federated (3 client + 1 server).
Implementation	Mengembangkan model federated dengan TensorFlow Federated di modul dinsos.py, dukcapil.py, kemenkes.py, kemudian di aggregasikan.
Testing	Melakukan uji unit & integrasi menggunakan test_all_models.py dan test_gabungan.py.

Deployment	Menyimpan model dalam folder saved_* dan mengaktifkan API Flask untuk inference.
Maintenance	Pembaruan aturan kelayakan atau retraining model dapat dilakukan tanpa mengganggu data instansi lain.

3. Requirement Gathering

A. Business Requirement

Tujuan utama proyek ini adalah membangun sistem penentuan kelayakan subsidi berbasis Federated Learning yang dapat memproses data lintas instansi tanpa menyalahi aturan privasi data.

Sistem ini diharapkan menjadi solusi *E-Government Data Integration* yang:

1. Menjaga keamanan & kerahasiaan data antar instansi pemerintah.
2. Meningkatkan akurasi keputusan kelayakan penerima bantuan sosial.
3. Mengurangi duplikasi data dan inkonsistensi antar lembaga.
4. Mendukung otomatisasi prediksi berbasis data riil dari DINSOS, DUKCAPIL, dan KEMENKES.
5. Memberikan hasil transparan yang dapat dijelaskan secara logis oleh aturan maupun hasil model.

Sasaran bisnis:

- Mempercepat validasi penerima subsidi.
- Meminimalkan human error dan bias manual.

B. User Requirement

Jenis Pengguna	Peran	Kebutuhan Utama
Operator Gabungan (Server)	Menjalankan federated training.	Menginisialisasi proses FedAvg dan memonitor metrik training.

Pengguna Umum / Pemerintah	Mengakses hasil prediksi.	Melihat apakah calon penerima layak atau tidak melalui API Flask atau dashboard.

C. Technical Requirement

Komponen	Spesifikasi
Bahasa Pemrograman	Python 3.10 / Python 3.11
Framework Machine Learning	TensorFlow, TensorFlow Federated (TFF)
Framework Web	Flask (untuk API prediksi)
Database / Storage	CSV-based dataset dan file SavedModel .pb
Library Pendukung	pandas, numpy, joblib, sklearn, pickle
Environment	Dapat berjalan di lokal (venv), WSL2, maupun Google Colab
Hardware Minimum	CPU ≥ 4 core, RAM ≥ 8 GB (opsional GPU untuk training cepat)
Output File	saved_model_gabungan_baru/, preprocess_gabungan_baru.pkl, JSON hasil prediksi.

D. Output Requirement

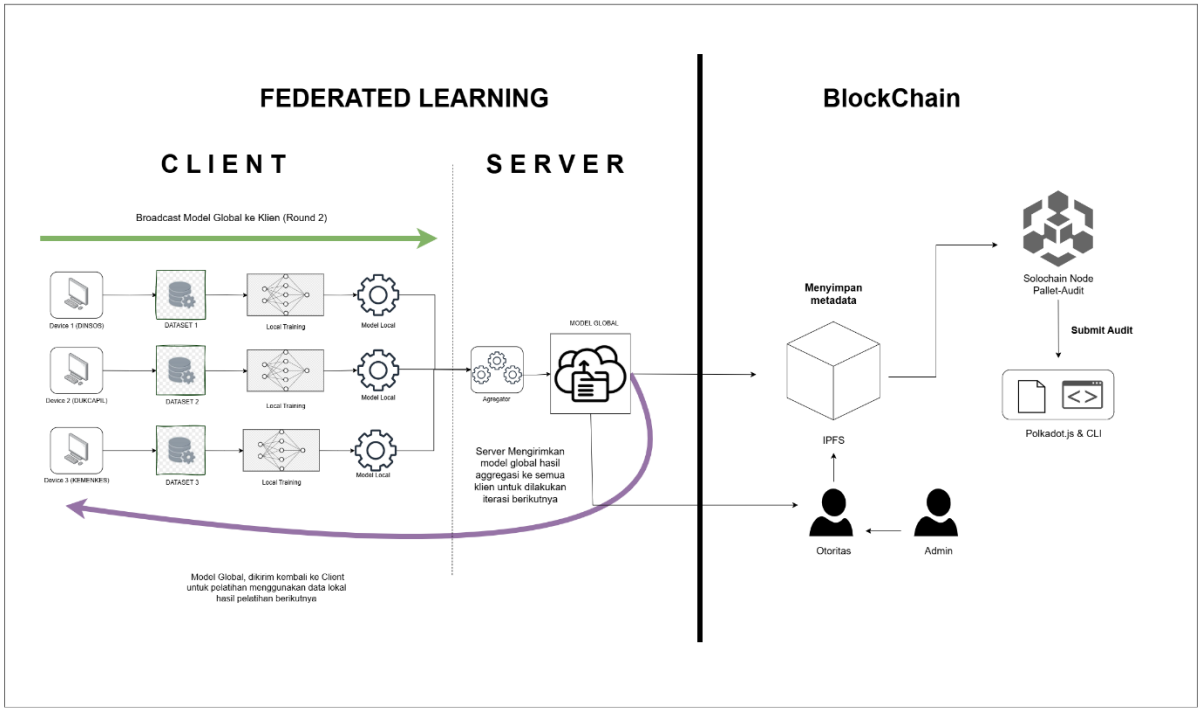
Hasil Prediksi API:

```json
{
    "prediksi": 1,
    "probabilitas": 0.94,
    "status": "Layak"
}
```

Hasil Training Model:

Tabel metrik per ronde federated learning (accuracy, loss).

WORKFLOW FEDERATED LEARNING



Alur Kerja Federated Learning (Subsidi Eligibility Prediction System)

1. Setiap klien memulai pelatihan lokal.
   Tiga lembaga (DINSOS, DUKCAPIL, dan KEMENKES) , masing-masing melatih model menggunakan dataset lokalnya sendiri tanpa berbagi data ke pihak lain.

2. Hasil pelatihan menghasilkan model lokal.
   Dari proses pelatihan tersebut, setiap lembaga menghasilkan Model Lokal (Local Model) yang sudah diperbarui sesuai karakteristik datanya.

3. Server melakukan proses agregasi (Federated Averaging).
   Di sisi server, komponen Aggregator menggabungkan semua model lokal menjadi satu Model Global Baru yang merepresentasikan hasil pembelajaran gabungan dari ketiga lembaga.

4. Server mengirimkan model global ke semua klien.
   Model global hasil agregasi kemudian dibroadcast ke semua lembaga untuk digunakan kembali dalam ronde pelatihan berikutnya (*Round 2*).

5. Klien melanjutkan pelatihan dengan model global.
   Masing-masing klien menerima model global tersebut, lalu melatih ulang menggunakan data lokalnya.

   Proses ini berulang — *local training* → *agregasi* → *broadcast kembali* — hingga model mencapai akurasi terbaik.

6. Terbentuk model global final.
   Setelah beberapa iterasi, terbentuk Model Global Akhir yang stabil dan representatif terhadap seluruh data nasional tanpa melanggar privasi antar lembaga.

Pencatatan ke Blockchain (Solochain Node – Pallet-Audit)

Setelah metadata disimpan di IPFS, hash-nya dikirim ke Blockchain layer, tepatnya ke Solochain Node menggunakan modul Pallet-Audit. Pallet ini berfungsi untuk mencatat riwayat model global yang dihasilkan di setiap iterasi — termasuk siapa kontributornya dan kapan model tersebut dibuat.

Proses ini dilakukan melalui Polkadot.js & CLI, yang mengirimkan transaksi *Submit Audit* ke Blockchain. Dengan cara ini, setiap versi model global yang dihasilkan akan memiliki jejak audit digital permanen di jaringan Blockchain, sehingga tidak bisa diubah, dihapus, atau dimanipulasi oleh pihak mana pun.