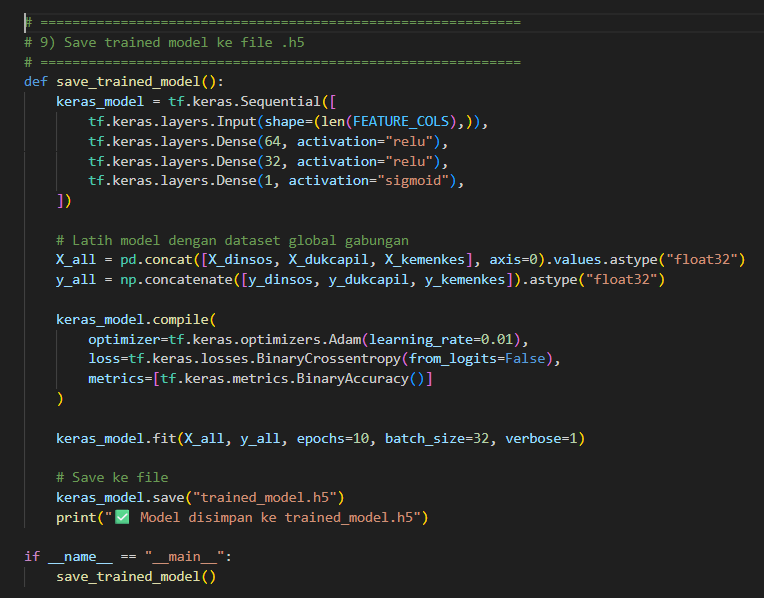
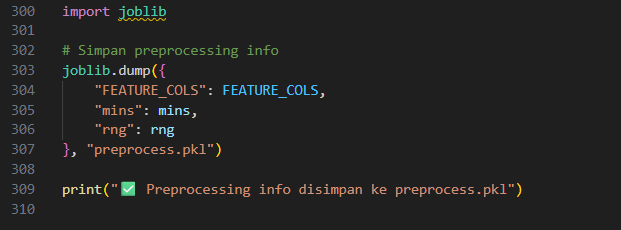
**Demo Aplikasi**

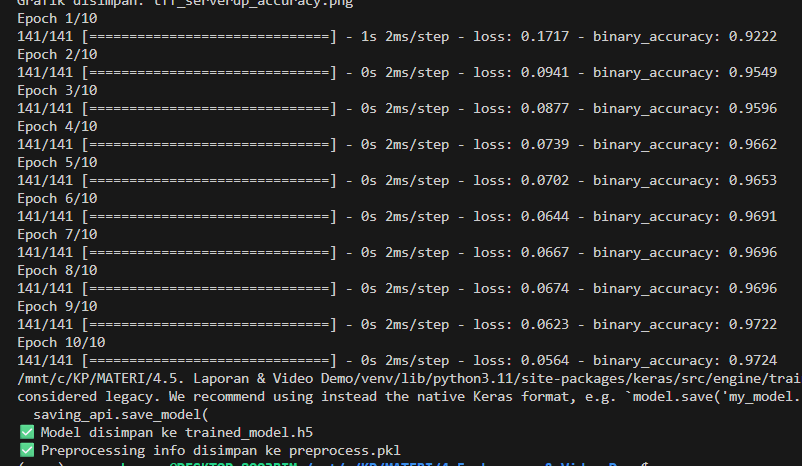


* Membuat arsitektur neural network sederhana (2 hidden layer).
* Menggabungkan dataset dari beberapa sumber.
* Melatih model dengan optimizer Adam + loss BinaryCrossentropy.
* Menyimpan model ke file trained\_model.h5 agar bisa digunakan lagi.



Kode ini menyimpan metadata preprocessing ke file preprocess.pkl menggunakan **joblib**. File ini nantinya akan dipakai bersamaan dengan model .h5 supaya pipeline prediksi berjalan mulus:

1. Load preprocess.pkl → terapkan normalisasi dengan mins dan rng.
2. Load trained\_model.h5 → lakukan prediksi pada data baru.



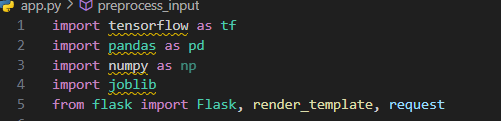
1. Model sudah dilatih **10 epoch** dengan hasil akurasi tinggi (**97.24%**).
2. File yang dihasilkan:

* **trained\_model.h5** → model machine learning yang siap dipakai.
* **preprocess.pkl** → metadata preprocessing untuk normalisasi data baru.

Menggunakan Flask untuk menjalankan atau mencoba model yang sudah dibuat :

**App.py**

1. Import Library



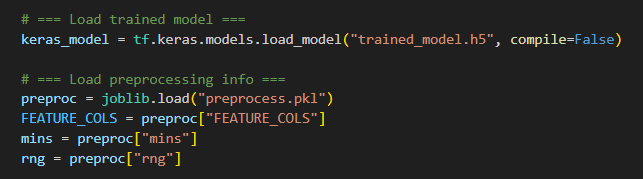
* **TensorFlow** → load model .h5 hasil training.
* **pandas, numpy** → manipulasi data dan preprocessing.
* **joblib** → load preprocessing info (preprocess.pkl).
* **Flask** → framework web Python untuk membuat form input & API prediksi.

1. Inisialisasi Flask



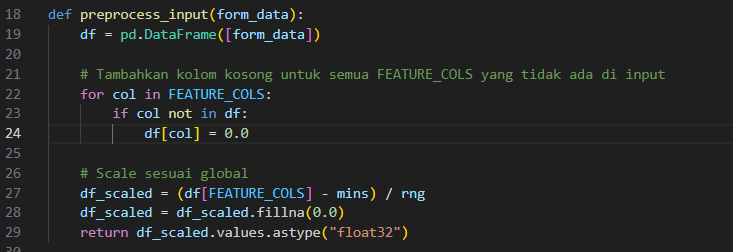
* Membuat aplikasi Flask

1. Load model dan Pre-Processing Info



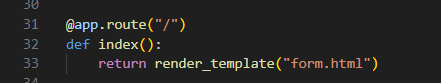
* **trained\_model.h5** → model hasil training.
* **preprocess.pkl** → menyimpan metadata preprocessing (nama kolom fitur, nilai minimum & range untuk normalisasi).
* compile=False → hanya load model untuk inference (tidak butuh compile ulang optimizer/loss).

1. Fungsi Pre-Processing Input



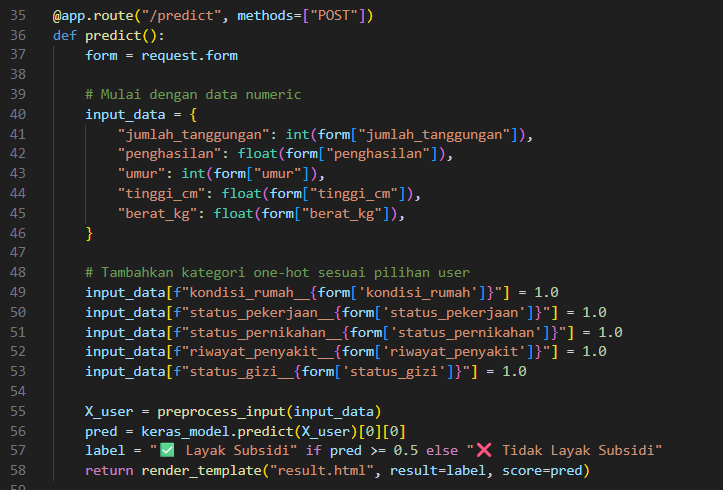
* Mengubah dictionary dari form menjadi DataFrame.
* Menambahkan kolom kosong untuk fitur yang hilang (0.0).
* Normalisasi sesuai parameter training (mins dan rng).
* Output berupa array float32 siap diprediksi.

1. Routing Halaman Utama



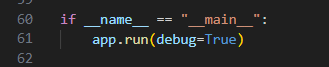
* Saat buka / (homepage), akan menampilkan form.html (form input user).

1. Routing Aplikasi

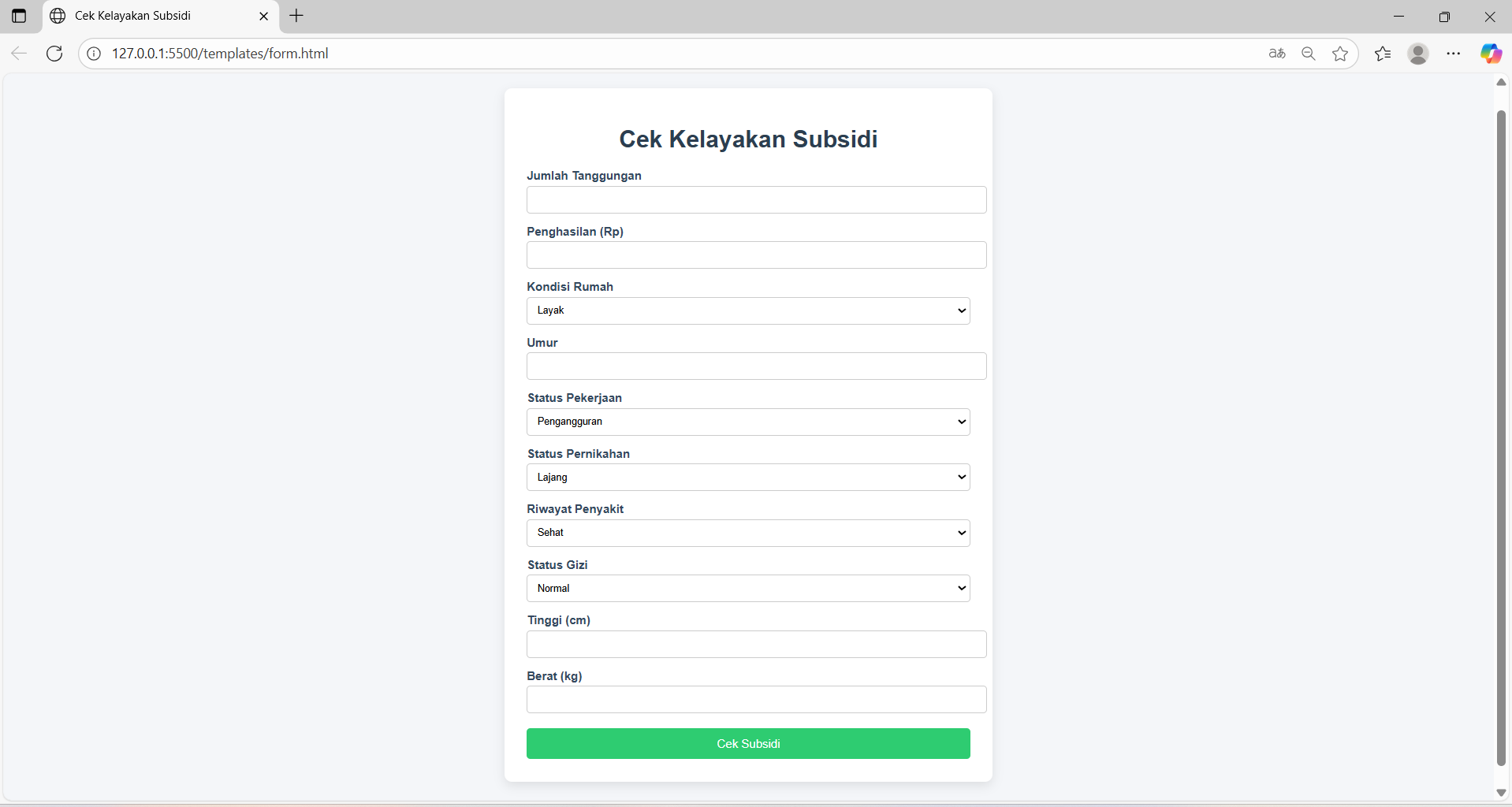


Alurnya:

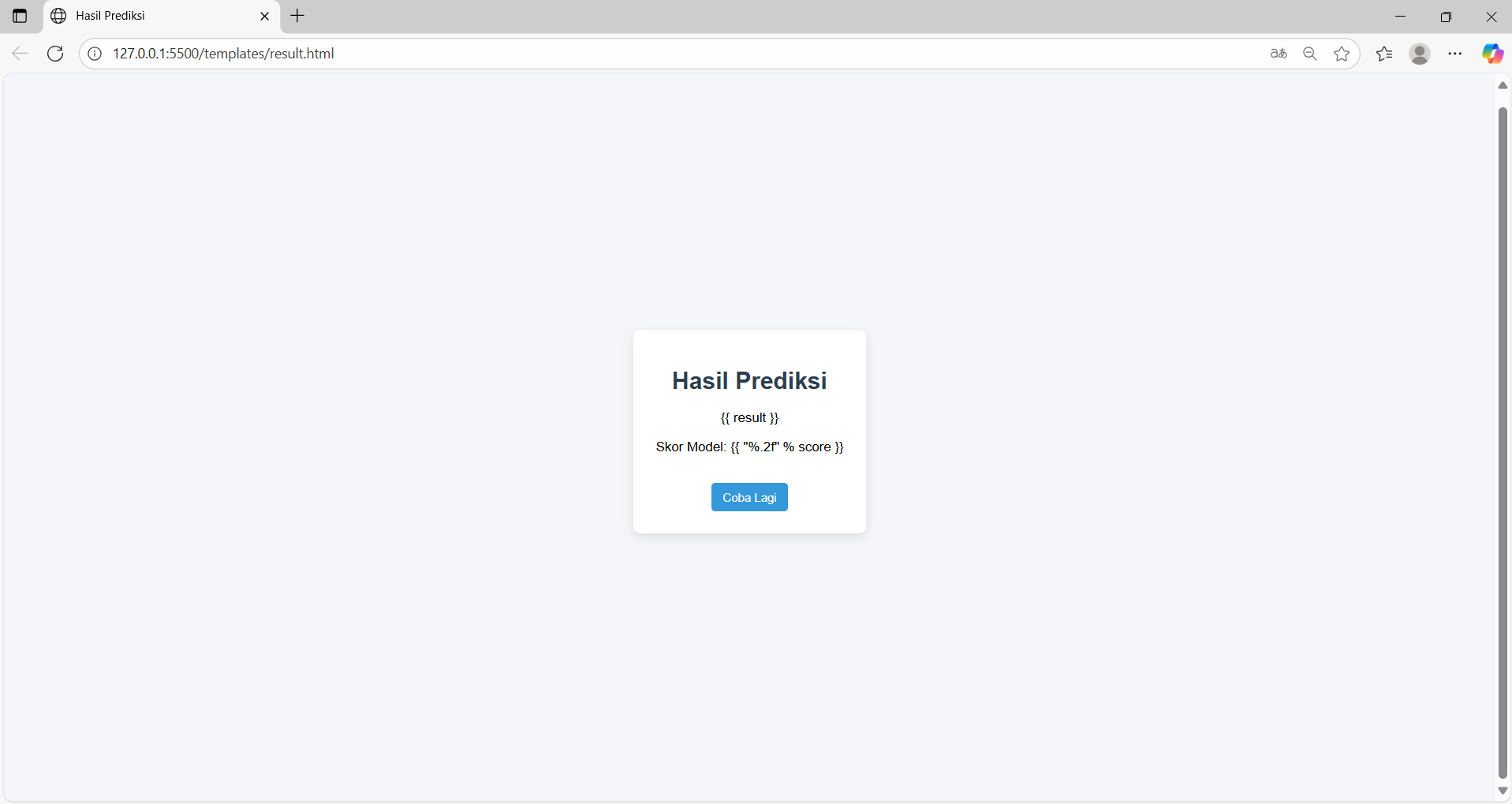
1. Ambil **input numerik** (tanggungan, penghasilan, umur, tinggi, berat).
2. Encode **kategori** menjadi one-hot (status, kondisi rumah, gizi, dll).
3. Preprocessing (normalize) → array float32.
4. Prediksi dengan model Keras.
5. Jika hasil >=0.5 → Layak Subsidi, kalau <0.5 → Tidak Layak.
6. Render hasil ke **result.html**.
7. Menjalankan Server



Form.html

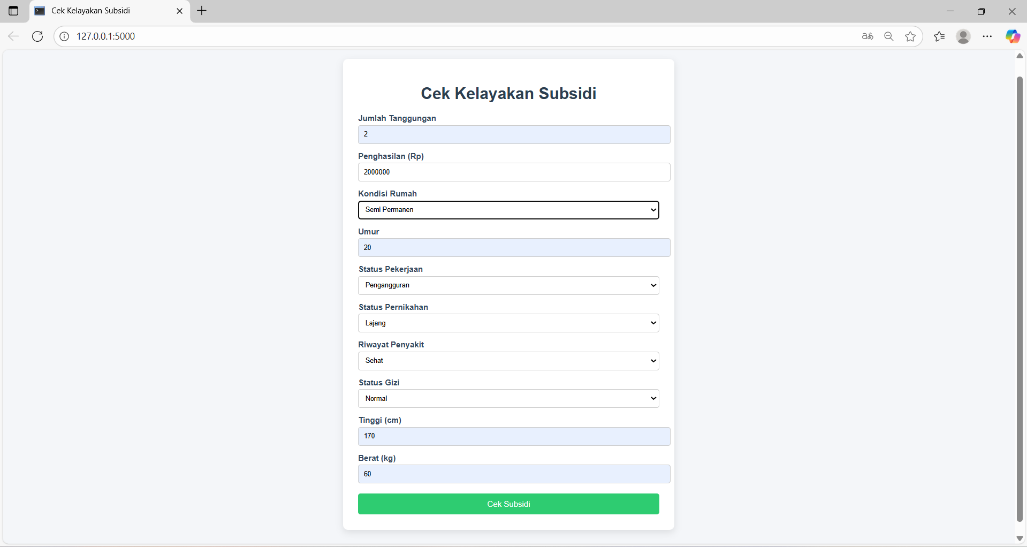
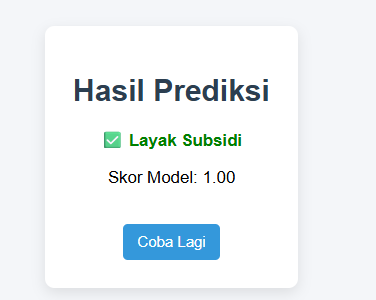


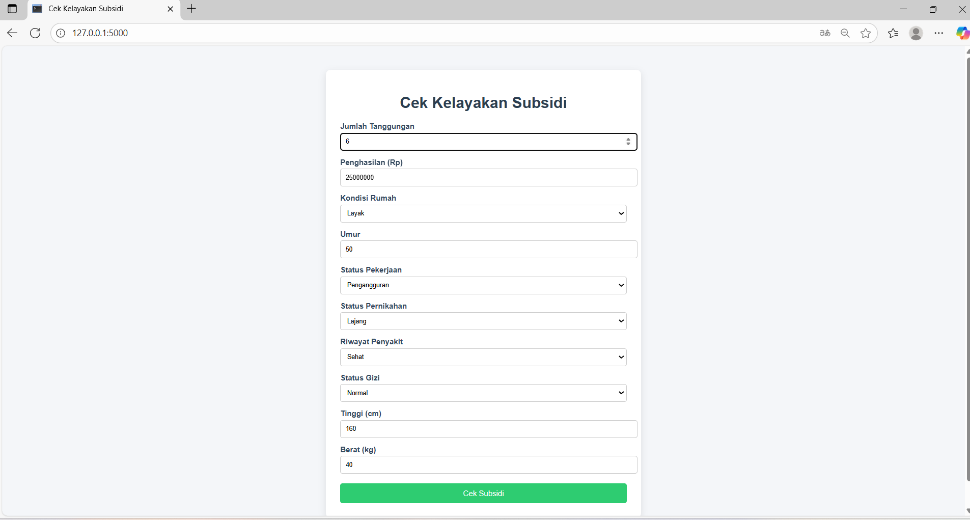
Result.html

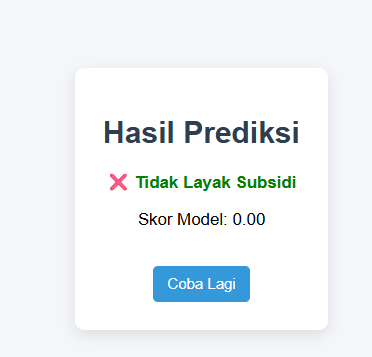


**UJI COBA**

1. **Kategori Layak**



1. **Kategori Tidak Layak**

****