1. **Arsitektur Umum**

|  |
| --- |
| **[User/Browser]**  **|**  **v**  **[Frontend (React.js)]**  **|**  **v**  **[Backend (Node.js/Express)]**  **|**  **v**  **[Machine Learning Model (Python/Flask/FastAPI)]**  **|**  **v**  **[Database (opsional)]** |

1. **Langkah Pembuatan**

**A. Frontend (React.js)**

* Buat halaman input untuk user (misal: usia, berat badan, tinggi badan, dll).
* Data dari user dikirim ke backend lewat API.
* Hasil prediksi atau rekomendasi dari backend ditampilkan ke user.

**B. Backend (Node.js/Express)**

* Menerima data dari frontend.
* Meneruskan data ke service machine learning (API Python).
* Menerima hasil prediksi dari service ML, lalu mengirimkan hasilnya ke frontend.

**C. Machine Learning (Python)**

* Model machine learning dibuat dengan library seperti scikit-learn, TensorFlow, atau PyTorch.
* Model di-deploy sebagai REST API menggunakan Flask atau FastAPI.
* Backend Node.js akan mengirim data ke API ini untuk mendapatkan hasil prediksi.

**D. Database (Opsional)**

* Jika ingin menyimpan data input user atau hasil prediksi, bisa tambahkan database seperti MongoDB atau PostgreSQL.

1. **Contoh Alur Kerja**
2. User mengisi form di website (React.js).
3. Data dikirim ke backend (Node.js).
4. Backend meneruskan data ke API ML (Python).
5. API ML memproses data dan mengirim hasil prediksi ke backend.
6. Backend mengirim hasil ke frontend untuk ditampilkan ke user.

**4. Contoh Kode Integrasi**

**A. Service Machine Learning (Python/Flask)**

|  |
| --- |
| **python** |
| from flask import Flask, request, jsonify  import pickle  app = Flask(\_\_name\_\_)  # Load model  model = pickle.load(open('model.pkl', 'rb'))  @app.route('/predict', methods=['POST'])  def predict():  data = request.get\_json()  # Contoh input: {'usia': 25, 'berat': 60, 'tinggi': 170}  input\_data = [data['usia'], data['berat'], data['tinggi']]  prediction = model.predict([input\_data])  return jsonify({'hasil': int(prediction[0])})  if \_\_name\_\_ == '\_\_main\_\_':  app.run(host='0.0.0.0', port=5000) |

* Jalankan dengan: python app.py
* API aktif di <http://localhost:5000/predict>

**B. Backend (Node.js/Express)**

|  |
| --- |
| **javascript** |
| const express = require('express');  const axios = require('axios');  const app = express();  app.use(express.json());  app.post('/api/predict', async (req, res) => {  try {  // Kirim data ke API Flask  const response = await axios.post('http://localhost:5000/predict', req.body);  // Kirim hasil ke frontend  res.json(response.data);  } catch (error) {  res.status(500).json({ error: 'Gagal memproses prediksi' });  }  });  app.listen(3000, () => {  console.log('Backend berjalan di http://localhost:3000');  }); |

**5. Tips Hosting/Deploy**

* Frontend: Vercel, Netlify (gratis & mudah).
* Backend: Render, Railway, atau VPS murah.
* Machine Learning API: Render, Railway, atau VPS.
* Database: MongoDB Atlas (gratis untuk skala kecil), ElephantSQL, dsb.

**Tips Tambahan**

* Gunakan format data JSON untuk komunikasi antar service.
* Pastikan port dan endpoint sudah benar.
* Untuk keamanan, bisa tambahkan autentikasi API.