LAPORAN RANCANGAN APLIKASI: PROBABILITAS PRO

Sistem Pakar Dinamis Metode Naive Bayes (Dynamic Expert System)

Status Dokumen: Final Blueprint
Target Waktu Pengerjaan: 1 Malam (Rapid Development)
Tech Stack: Laravel 11, Inertia.js (Vue 3), TailwindCSS, MySQL.

1. PENDAHULUAN & KONSEP

Aplikasi ini bernama **"ProbabilitasPro"**. Merupakan platform berbasis web (SPA) untuk membangun sistem pakar diagnosa (studi kasus: Kerusakan Komputer) menggunakan metode *Naive Bayes Classifier*. Sistem ini dirancang agnostik, artinya pengguna dapat mendefinisikan sendiri variabel Gejala (Predictor) dan Kerusakan (Class) secara dinamis.

Filosofi Desain

- **Tema Visual:** *Light Blue Professional* (Warna dominan: Slate-50, Sky-500, Blue-600).
- **UX:** *Infinite Canvas* (Zoom/Pan) untuk manajemen knowledge base, dan *Interactive Drawer* untuk proses inferensi (analisis).

2. ARSITEKTUR SISTEM & PILIHAN TEKNOLOGI

Untuk mencapai kebutuhan SPA yang reaktif dengan backend Laravel, arsitektur berikut **WAJIB** digunakan:

- **Backend:** Laravel 11 (API & Routing).
- **Frontend Framework:** Vue.js 3 (Composition API) via **Inertia.js**.
    - *Alasan:* Memenuhi syarat SPA dan Ajax, namun coding style tetap "Monolith" yang mudah dipahami.
- **State Management:** Pinia (Untuk mengelola state zoom, posisi tabel, dan data analisis).
- **CSS Framework:** TailwindCSS (Utility-first).
- **Icon:** Lucide-Vue atau Heroicons.

3. TERMINOLOGI AKADEMIK (MAPPING)

Agar sesuai dengan standar akademis dalam UI dan Database:

Bahasa User (Awam)	**Istilah Akademis / Teknis (Sistem)**	**Simbol Matematika**
Judul	Domain Permasalahan	-
Variabel X (Gejala)	Fitur Prediktor (*Predictor*	$X = \{x_1, x_2, ..., x_n\}$

	Features / Evidence)	
Variabel Y (Kerusakan)	Label Kelas (*Class Labels / Hypothesis*)	$C = \{c_1, c_2, ..., c_m\}$
Analisis	Inferensi Probabilistik (*Probabilistic Inference*)	$P(C_k

4. SKEMA DATABASE (MIGRATION)

Hanya membutuhkan 4 tabel utama untuk sistem dinamis ini.

A. Table: projects

Menyimpan sesi analisis/studi kasus.

```
Schema::create('projects', function (Blueprint $table) {
    $table->id();
    $table->string('title'); // Judul Studi Kasus
    $table->text('description')->nullable(); // Deskripsi Masalah
    $table->string('x_label')->default('Gejala'); // Nama kustom untuk Variabel X
    $table->string('y_label')->default('Kerusakan'); // Nama kustom untuk Variabel Y
    $table->timestamps();
});
```

B. Table: attributes (Variabel X / Gejala)

```
Schema::create('attributes', function (Blueprint $table) {
    $table->id();
    $table->foreignId('project_id')->constrained()->cascadeOnDelete();
    $table->string('code'); // Contoh: G001, X1
    $table->string('name'); // Contoh: Lampu Indikator Mati
    $table->timestamps();
});
```

C. Table: classes (Variabel Y / Kerusakan)

```
Schema::create('classes', function (Blueprint $table) {
    $table->id();
    $table->foreignId('project_id')->constrained()->cascadeOnDelete();
    $table->string('code'); // Contoh: K001, Y1
    $table->string('name'); // Contoh: IC Power Rusak
    $table->double('prior_probability')->default(0); // P(Y) - Opsional jika ingin set manual
    $table->timestamps();
});
```

D. Table: training_data (Rules/Matrix)

Tabel ini menyimpan relasi (checkbox) pada Matrix UI.

```
Schema::create('training_data', function (Blueprint $table) {
    $table->id();
    $table->foreignId('project_id')->constrained()->cascadeOnDelete();
    $table->foreignId('class_id')->constrained('classes')->cascadeOnDelete(); // Y
    $table->foreignId('attribute_id')->constrained('attributes')->cascadeOnDelete(); // X
    $table->boolean('is_associated')->default(false); // Checkbox value
    // Jika true, maka Gejala X muncul pada Kerusakan Y
    $table->timestamps();

    // Unique constraints agar tidak duplikat pasangan X-Y
    $table->unique(['class_id', 'attribute_id']);
});
```

5. ALUR APLIKASI & DESAIN UI (DETIL)

Halaman 1: Dashboard (Landing)

- **Layout:** Card Grid modern.
- **Content:** List projects (Recent Analysis). Menampilkan Judul, Tanggal, dan jumlah variabel.
- **Fab/Button:** "Inisiasi Studi Kasus Baru".
- **Modal Form (Saat Button diklik):**
    - Input: Judul Studi Kasus.
    - Input: Deskripsi.
    - Input Group X: Label (default: Gejala), Estimasi Jumlah Awal (Input Number).
    - Input Group Y: Label (default: Kerusakan), Estimasi Jumlah Awal (Input Number).
    - Action: "Generate Workspace".

Halaman 2: The Workspace (Inti Aplikasi)

Halaman ini dibagi menjadi 3 layer interaktif.

A. Sidebar Kiri (Konfigurasi Variabel) - *Collapsible*

- **Width:** 25% (w-1/4), bisa di-toggle tutup/buka.
- **Tab:** Switch antara tab "Prediktor (X)" dan "Kelas (Y)".
- **List Item:** Input text untuk mengedit nama variabel (misal: ubah "X1" jadi "Layar Blue Screen").
- **Reactivity:** Saat nama diubah di sini, Header/Label di Tabel Utama langsung berubah.
- **Button:** "+ Tambah Variabel" di bawah list.

B. Center Stage (Matrix Table) - *Infinite Canvas*

- **Container:** overflow-hidden relative.
- **Interaksi:**
  - **Drag:** Mouse down + move untuk menggeser area pandang tabel.
  - **Zoom:** Ctrl + Scroll atau menggunakan **Slider Range** di pojok kanan bawah (Scale 0.5x s/d 2.0x).
- **Komponen Tabel:**
  - **Header Atas (Columns):** Merepresentasikan **Variabel Y (Kelas/Kerusakan)**.
  - **Header Kiri (Rows):** Merepresentasikan **Variabel X (Prediktor/Gejala)**.
  - **Cell (Pertemuan X & Y):** Checkbox Besar.
    - *Logika:* Jika user mencentang Cell(Row=G1, Col=K1), artinya "Gejala 1 sangat mungkin terjadi pada Kerusakan 1".
  - **Tooltip:** Hover pada header X/Y memunculkan deskripsi lengkap.

C. Bottom Drawer (Analisis Layer) - *Draggable*

- **Posisi:** Fixed bottom, default terlipat (hanya terlihat handle barnya).
- **Interaksi:** Bisa ditarik ke atas (max 50vh) atau di-minimize.
- **Konten:**
  - **Bagian Input:** Daftar Checklist dinamis dari semua Variabel X (Gejala) yang sudah didefinisikan. User mencentang apa yang dialami saat ini.
  - **Tombol Action:** "Lakukan Inferensi Probabilistik".
  - **Efek Visual:** Saat diklik, overlay muncul dengan animasi *Aurora Borealis* (CSS Gradient Animation) + partikel bintang (CSS/JS Canvas), loading 2 detik.
  - **Bagian Output (Result):**
    - Menampilkan Card hasil diagnosa.
    - Mengurutkan Kerusakan (Y) dari persentase tertinggi ke terendah.
    - Format: "Berdasarkan gejala yang dipilih, probabilitas kerusakan adalah: **IC Power (87%)**".

6. LOGIKA NAIVE BAYES (IMPLEMENTASI)

Karena input user berupa "Checkbox" (Relasi biner: Terkait atau Tidak), kita menggunakan pendekatan **Bernoulli Naive Bayes** atau simplifikasi probabilitas kondisional.

Algoritma Backend (Controller):

1. **Input:** Array gejala yang dipilih user (misal: $X_{selected} = \{x_1, x_3\}$).
2. Proses:
   Untuk setiap Kelas Y_k (Kerusakan):
   - Hitung Likelihood (L):
     Ambil data dari tabel training_data.
     - Jika checkbox (x_i, Y_k) dicentang di tabel utama, beri bobot probabilitas tinggi (misal 0.9).
     - Jika tidak dicentang, beri bobot rendah/epsilon (misal 0.1 - *Laplace*

Smoothing agar tidak nol).
    - Rumus Sederhana:
      $$Score(Y_k) = P(Y_k) \times \prod_{i \in selected} P(x_i | Y_k)$$

      (Dimana $P(x_i|Y_k)$ adalah 0.9 jika dicentang, 0.1 jika tidak).
3. Normalisasi:
   Ubah Score menjadi Persentase.
   $$Persentase(Y_k) = \frac{Score(Y_k)}{\sum Score(All\_Y)} \times 100\%$$

7. INSTRUKSI VISUAL (TAILWIND CSS)

- **Warna Dasar:** bg-slate-50
- **Sidebar:** bg-white border-r border-slate-200
- **Table Headers:** bg-sky-100 text-sky-900 font-bold
- **Checkbox:** Gunakan custom style. Warna centang text-blue-600.
- **Magic Loading:**
  ```
  @keyframes aurora {
    0% { background-position: 0% 50%; }
    50% { background-position: 100% 50%; }
    100% { background-position: 0% 50%; }
  }
  .magic-loader {
    background: linear-gradient(270deg, #3b82f6, #8b5cf6, #06b6d4);
    background-size: 600% 600%;
    animation: aurora 3s ease infinite;
  }
  ```

8. LANGKAH PENGERJAAN (UNTUK AI/DEV)

1. **Setup Laravel:** Install Laravel 11 + Breeze (Vue/Inertia).
2. **Database:** Buat migration sesuai skema di poin 4.
3. **Models:** Buat Model Project, Attribute, Class, TrainingData. Setup relasi hasMany dan belongsTo.
4. **API/Controller:**
    - ProjectController: CRUD project.
    - WorkspaceController: Handle update nama variabel (Ajax), update checkbox (Ajax), dan fungsi analyze().
5. **Frontend (Vue Components):**
    - SidebarConfig.vue: Form input list.
    - MatrixTable.vue: Logika zoom (CSS transform scale) dan drag (event listeners mousemove).
    - AnalysisDrawer.vue: Bottom sheet logic dan animasi hasil.
6. **Integrasi:** Sambungkan state checkbox tabel ke database via debounced API call (agar

auto-save).

End of Document