

# MinRisk Solution Specification Document

Version 1.0 | Technical Architecture & System Design

## Table of Contents

- [Executive Summary](#)
- [System Overview](#)
- [Architecture](#)
- [Core Modules](#)
- [Data Model](#)
- [Security](#)
- [AI/ML Capabilities](#)
- [Integration Points](#)
- [Deployment](#)
- [Performance](#)

## 1. Executive Summary

### 1.1 Product Overview

**MinRisk** is an enterprise-grade Risk Management platform designed to help organizations identify, assess, mitigate, and monitor risks across their operations. The platform combines traditional risk management frameworks with AI-powered capabilities for intelligent risk identification and continuous monitoring.

### 1.2 Key Differentiators

Feature	Description
AI Risk Intelligence	Automated scanning of external events with AI-powered relevance analysis
DIME Control Framework	Structured control assessment methodology
Real-time KRI Monitoring	Threshold-based alerting with escalation workflows
Incident-Risk Mapping	AI-assisted linking of incidents to root cause risks
Multi-tenant Architecture	Secure organization isolation via Row-Level Security

### 1.3 Target Users

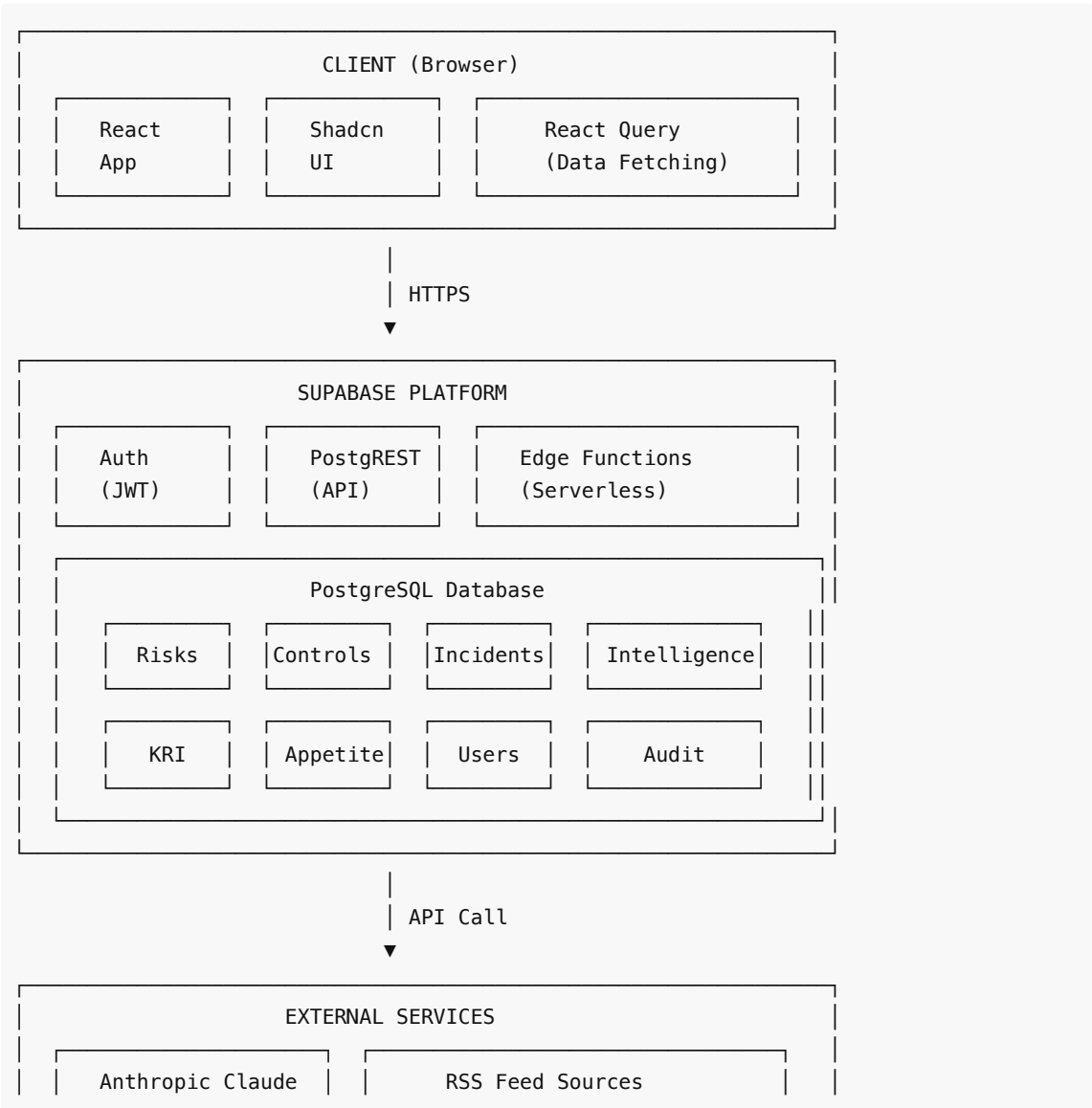
- Chief Risk Officers (CRO)
- Risk Managers
- Compliance Officers
- Internal Auditors
- Business Unit Heads
- IT Security Teams

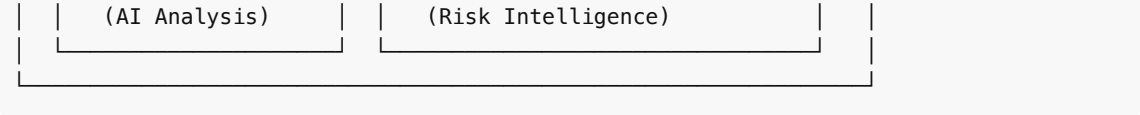
## 2. System Overview

2.1 Technology Stack

Layer	Technology
Frontend	React 18, TypeScript, Vite
UI Components	Shadcn/UI, Radix Primitives, TailwindCSS
State Management	React Query, Zustand
Backend	Supabase (PostgreSQL + Edge Functions)
Authentication	Supabase Auth (JWT-based)
AI Services	Anthropic Claude API
Hosting	Render (Production)

2.2 System Diagram





### 3. Architecture

#### 3.1 Frontend Architecture

```
src/
├── components/           # UI Components
│   ├── admin/           # Administration panels
│   ├── ai/              # AI Risk Generator
│   ├── analytics/       # Charts & Reports
│   ├── auth/            # Login/Signup
│   ├── controls/        # Control management
│   ├── dashboard/       # Main dashboard
│   ├── incidents/       # Incident management
│   ├── kri/             # KRI/KCI module
│   ├── riskIntelligence/ # External events & alerts
│   ├── risks/           # Risk register
│   └── ui/              # Reusable UI primitives
├── lib/                 # Service layer
│   ├── risks.ts         # Risk CRUD operations
│   ├── controls.ts      # Control operations
│   ├── incidents.ts     # Incident operations
│   ├── kri.ts           # KRI management
│   ├── riskIntelligence.ts # External event scanning
│   ├── ai.ts            # AI service calls
│   ├── appetiteTolerance.ts # Appetite framework
│   ├── auth.ts          # Authentication
│   └── taxonomy.ts      # Category management
├── types/               # TypeScript type definitions
└── hooks/               # Custom React hooks
```

#### 3.2 Backend Architecture

**Supabase Edge Functions:**

Function	Purpose
call-ai	Proxy for Claude API calls
analyze-intelligence	AI relevance analysis for events
invite-user	User invitation email sending

**Database Functions:**

Function	Purpose
generate_risk_code()	Sequential risk code generation

generate_control_code()	Sequential control code generation
generate_kri_code()	Sequential KRI code generation
calculate_residual_risk()	Dynamic residual calculation
accept_ai_suggestion()	Incident-risk mapping workflow

### 3.3 Row-Level Security (RLS)

All tables implement organization-scoped RLS:

```
-- Example RLS Policy
CREATE POLICY "Users can see their organization's risks"
ON risks FOR SELECT
USING (
  organization_id = (
    SELECT organization_id FROM user_profiles
    WHERE id = auth.uid()
  )
);
```

## 4. Core Modules

### 4.1 Risk Management Module

**Capabilities:**

- Risk register with full CRUD operations
- Auto-generated risk codes (DIV-CAT-001 format)
- Inherent/Residual risk scoring
- Risk-control linkage (Many-to-Many)
- Risk-KRI linkage (Many-to-Many)
- Period-based risk snapshots
- Priority flagging

**Risk Lifecycle:**



### 4.2 Control Management Module

**Capabilities:**

- Control library with CRUD
- DIME framework scoring
- Control-risk linkage
- Control effectiveness calculation
- Control type classification (Preventive/Detective/Corrective)
- Target dimension (Likelihood/Impact/Both)

#### DIME Calculation:

```
function calculateControlEffectiveness(  
  design: number,      // 0-3  
  implementation: number, // 0-3  
  monitoring: number,  // 0-3  
  evaluation: number   // 0-3  
): number {  
  // Special case: If not designed or implemented, effectiveness = 0  
  if (design === 0 || implementation === 0) return 0;  
  
  return (design + implementation + monitoring + evaluation) / 12;  
}
```

#### Residual Risk Calculation:

```
function calculateResidualRisk(  
  inherent: number,  
  maxEffectiveness: number  
): number {  
  return Math.max(1, inherent - Math.round((inherent - 1) * maxEffectiveness));  
}
```

### 4.3 Incident Management Module

#### Capabilities:

- Incident logging with auto-generated codes (INC-001)
- Status workflow management
- Root cause documentation
- Incident-to-risk mapping (manual + AI-assisted)
- Description amendment tracking
- Audit trail for all changes

#### AI-Assisted Mapping:

```
interface AIRiskSuggestion {  
  risk_id: string;  
  risk_code: string;  
  confidence_score: number; // 0-100  
  keywords: string[];  
  reasoning: string;  
  link_type_suggestion: string;  
}
```

### 4.4 KRI/KCI Module

#### Capabilities:

- KRI definition management
- Multiple indicator types (Leading/Lagging/Concurrent)

- Configurable thresholds (Green/Yellow/Red)
- Data entry with quality flags
- Automatic alert creation on breach
- Alert lifecycle management

#### Threshold Logic:

Direction	Green	Yellow	Red
Above	Value $\leq$ Lower	Lower < Value $\leq$ Upper	Value > Upper
Below	Value $\geq$ Upper	Lower $\leq$ Value < Upper	Value < Lower
Between	Lower $\leq$ Value $\leq$ Upper	Near boundaries	Outside range

## 4.5 Risk Intelligence Module

#### Capabilities:

- RSS feed source management
- External event ingestion
- Duplicate event detection
- AI-powered relevance analysis
- Risk impact scoring suggestions
- Alert treatment workflow
- Treatment audit logging

#### Relevance Analysis Flow:

```

External Event
    ↓
[AI Relevance Check] ← Organization's Risks
    ↓
For each relevant risk:
  - Confidence score (0-100%)
  - Likelihood change suggestion (-2 to +2)
  - Impact change suggestion (-2 to +2)
  - Reasoning
    ↓
Intelligence Alert Created
    ↓
Admin Review → Accept/Reject
    ↓
Risk scores updated (if accepted)
  
```

## 4.6 Risk Appetite & Tolerance Module

#### Capabilities:

- Version-controlled appetite statements
- Category-level appetite definitions
- Tolerance metrics with multiple types
- Breach detection and tracking
- Board acceptance workflow

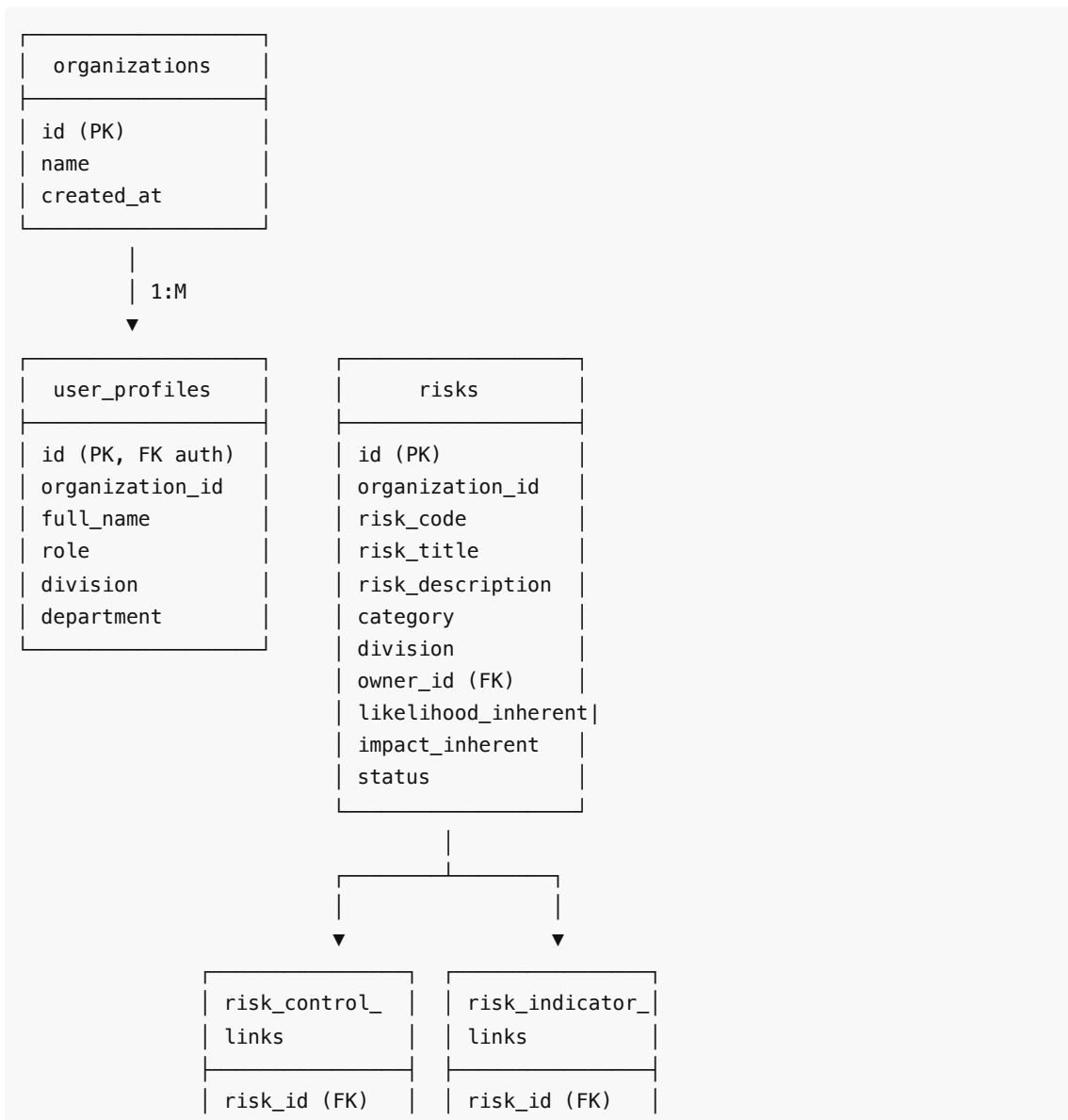
- Temporary exception management

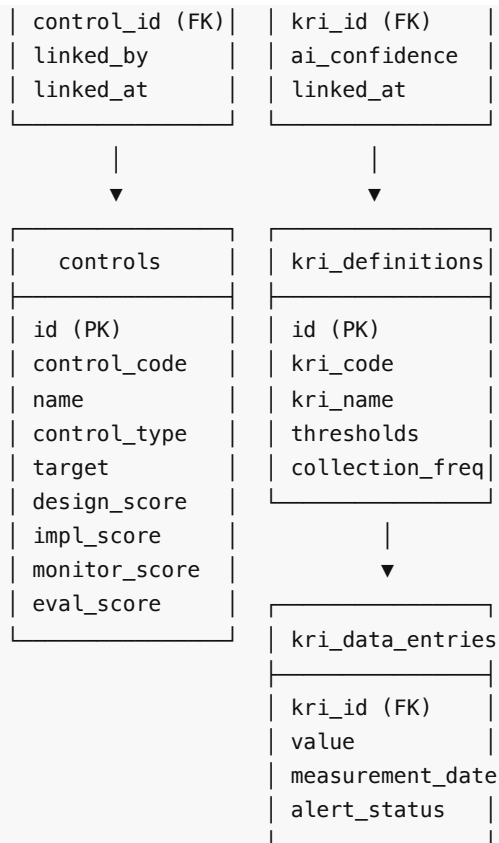
#### Metric Types:

Type	Logic
RANGE	Value must be within min-max bounds
MAXIMUM	Value must not exceed threshold (lower is better)
MINIMUM	Value must meet minimum floor (higher is better)
DIRECTIONAL	Rate of change over lookback period

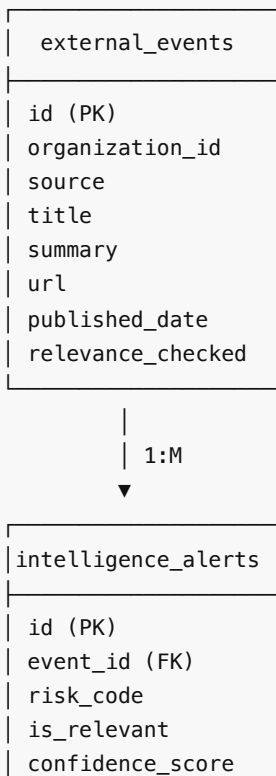
## 5. Data Model

### 5.1 Core Tables



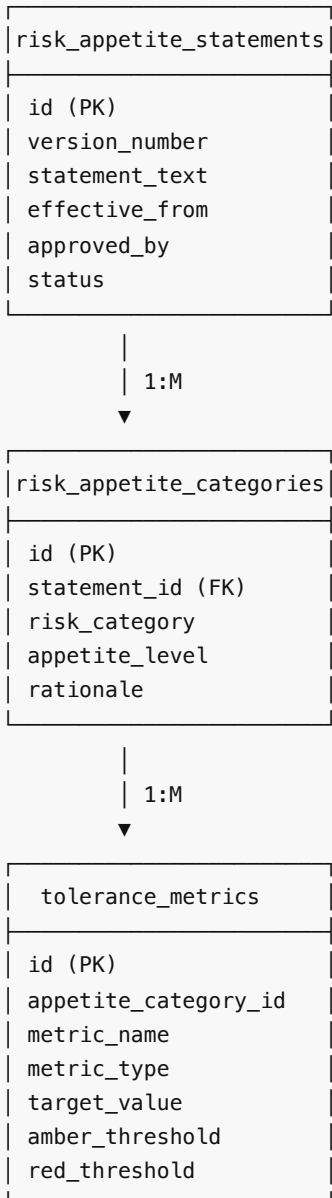


## 5.2 Intelligence Tables



likelihood_change
impact_change
status
reviewed_by

## 5.3 Appetite Tables



## 6. Security

### 6.1 Authentication

- **Provider:** Supabase Auth
- **Method:** Email/Password + Magic Links

- **Session:** JWT tokens with refresh
- **Expiry:** 1 hour access, 7 day refresh

### 6.2 Authorization

**Role-Based Access Control:**

Role	Risks	Controls	Incidents	Admin
Admin	Full	Full	Full	Full
Risk Manager	CRUD	CRUD	CRUD	Read
Viewer	Read	Read	Read	None

### 6.3 Row-Level Security

All tables implement RLS ensuring:

- Users only see their organization's data
- Organization context derived from `auth.uid()`
- No cross-organization data leakage

### 6.4 API Security

- All API calls require valid JWT
- Edge Functions validate auth before processing
- External API keys (Anthropic) stored as Supabase secrets
- No sensitive keys exposed to client

---

## 7. AI/ML Capabilities

### 7.1 AI Provider

- **Model:** Anthropic Claude 3.5 Sonnet
- **Integration:** Via Supabase Edge Function proxy
- **Rate Limiting:** Built-in via Supabase

### 7.2 AI Features

Feature	Description	Input	Output
Risk Generation	Create risks from context	Industry, unit, category	Risk objects
Risk Classification	Classify user statement	Free text, taxonomy	Category + refined statement
Statement Refinement	Polish risk description	Draft statement	Professional statement
Control Recommendations	Suggest controls	Risk details	Control objects with DIME
Incident Mapping	Match incidents to risks	Incident, risk list	Suggestions with confidence

Event Relevance	Analyze external events	Event, risks	Relevance scores + changes
-----------------	-------------------------	--------------	----------------------------

### 7.3 Demo Mode

When AI is unavailable or for testing:

- `VITE_AI_DEMO_MODE=true` enables mock responses
- Realistic sample data returned
- No API charges incurred

## 8. Integration Points

### 8.1 Inbound Integrations

Source	Method	Data
RSS Feeds	Scheduled scan	External events
Manual CSV	File upload	Bulk risk import

### 8.2 Outbound Integrations

Target	Method	Data
Email (SMTP)	Supabase edge	User invitations
Reports	Export	CSV/PDF reports

### 8.3 API Endpoints

All data operations via Supabase PostgREST:

```
Base URL: https://<project>.supabase.co/rest/v1

GET    /risks      # List risks
POST   /risks      # Create risk
PATCH /risks?id=eq.X # Update risk
DELETE /risks?id=eq.X # Delete risk

# Similar patterns for:
/controls, /incidents, /kri_definitions, /external_events, etc.
```

## 9. Deployment

### 9.1 Production Environment

Component	Platform	URL
Frontend	Render	new-minrisk-production-dec.onrender.com
Database	Supabase	*.supabase.co

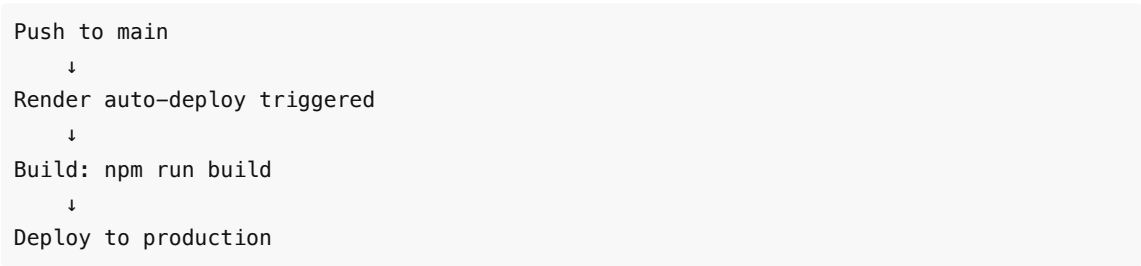
Edge Functions	Supabase	*.functions.supabase.co
----------------	----------	-------------------------

9.2 Environment Variables

```
# Required
VITE_SUPABASE_URL=https://xxx.supabase.co
VITE_SUPABASE_ANON_KEY=xxx

# Optional
VITE_AI_DEMO_MODE=false # Enable mock AI responses
```

9.3 CI/CD Pipeline



9.4 Database Migrations

Migrations stored in /supabase/migrations/ :

- Numbered by date: 20250101\_feature\_name.sql
- Applied via Supabase CLI: supabase db push
- Optional seeds: OPTIONAL\_\*.sql
- Disabled migrations: DISABLED\_\*.sql

10. Performance

10.1 Optimization Strategies

Area	Strategy
Database	Indexes on foreign keys, RLS-optimized queries
Frontend	React Query caching, lazy loading
API	Pagination (default 100 rows)
Assets	Vite code splitting, static serving

10.2 Scalability

Tier	Users	Risks	Notes
Free	1-10	500	Supabase free tier

Pro	11-100	5,000	Supabase Pro
Enterprise	100+	Unlimited	Custom deployment

### 10.3 Monitoring

- **Database:** Supabase Dashboard → Logs
- **Frontend:** Browser DevTools
- **Errors:** Console logging (future: Sentry integration)

---

## Appendix A: Risk Categories

Default taxonomy:

Category	Sub-categories
Credit Risk	Counterparty, Concentration, Settlement
Market Risk	Interest Rate, FX, Equity
Operational Risk	Process, People, Systems, External
Liquidity Risk	Funding, Market Liquidity
Compliance Risk	Regulatory, Legal, Conduct
Strategic Risk	Business Model, Reputation, Political
Technology Risk	Cyber, IT Infrastructure, Data

---

## Appendix B: Glossary

Term	Definition
<b>DIME</b>	Design, Implementation, Monitoring, Evaluation
<b>KRI</b>	Key Risk Indicator
<b>KCI</b>	Key Control Indicator
<b>RAS</b>	Risk Appetite Statement
<b>RLS</b>	Row-Level Security
<b>Inherent Risk</b>	Risk before controls
<b>Residual Risk</b>	Risk after controls