KYB Tool - Feature Specifications Document

Part 1: Core Features and User Stories

Document Information

- **Document Type**: Feature Specifications Document
- **Project**: KYB Tool - Enterprise-Grade Know Your Business Platform
- **Version**: 1.0
- **Date**: January 2025
- **Status**: Final Specification
- **Pages**: Part 1 of 4

---

1. Core Feature Overview

1.1 Feature Prioritization Matrix

Based on the Kano Model analysis and customer research, features are categorized as follows:

Must-Have Features (Phase 1 - MVP)

- Business Classification Engine
- Risk Assessment System
- Compliance & Sanctions Screening
- Web Dashboard
- RESTful API with Authentication
- Basic Reporting and Export

Performance Features (Phase 2-3)

- Advanced AI and Predictive Analytics
- Real-time Monitoring and Alerts
- Comprehensive SDK Ecosystem
- Advanced Dashboard and Analytics
- Multi-region Support

Attractive Features (Phase 3-4)

- Conversational AI Interface
- Computer Vision Document Analysis
- Blockchain and Web3 Support
- Industry-specific Vertical Solutions
- Open API Marketplace

1.2 Feature Dependencies Map

```
Business Classification Engine
├── Data Ingestion Service
├── ML Models (BERT + XGBoost)
├── Code Database (MCC/NAICS/SIC)
└── Confidence Scoring

Risk Assessment System
├── Business Classification Engine
├── Website Analysis Service
├── Sanctions Screening
├── Predictive ML Models
└── Risk Factor Database

Web Dashboard
├── Authentication Service
├── Business Management
├── Risk Visualization
├── Report Generation
└── Settings Management

API Gateway
├── Authentication Service
├── Rate Limiting
├── Request Routing
├── Response Caching
└── Monitoring Integration
```

---

2. Epic 1: Business Classification Engine

2.1 Epic Overview

Epic Description: Automated business classification system that analyzes business descriptions, websites, and other data sources to assign accurate industry codes (MCC, NAICS, SIC) with confidence scores.

Business Value: Reduces manual classification time from 15-30 minutes to under 2 seconds while achieving 95%+ accuracy, enabling automated onboarding and risk assessment.

Success Metrics:

- Classification accuracy: ≥95% for primary codes
- Response time: <2 seconds (95th percentile)
- Confidence score calibration: 90% of high-confidence predictions are correct
- Coverage: Support for 1000+ MCC codes, 2000+ NAICS codes, 1000+ SIC codes

2.2 User Stories

Story 1.1: Basic Business Classification

As a payment processor integration developer
I want to submit business information and receive industry code classifications
So that I can automatically categorize merchants during onboarding

Acceptance Criteria:

```gherkin
```

```
Given I have valid API credentials and business information
When I submit a POST request to /api/v1/classify with business data
Then I should receive a response within 2 seconds
And the response should include MCC, NAICS, and SIC codes
And each code should have a confidence score between 0.0 and 1.0
And the response should indicate which code is the primary classification
And the API should return appropriate error messages for invalid input

Scenario: Successful classification
Given business description "Online retail clothing store selling fashion apparel"
When I call the classification API
Then I should receive MCC code "5691" (Women's Ready-to-Wear Stores)
And confidence score should be > 0.85
And NAICS code should be "448120" (Women's Clothing Stores)
And SIC code should be "5621" (Women's Ready-to-Wear Stores)

Scenario: Ambiguous classification
Given business description "Consulting services"
When I call the classification API
Then I should receive multiple code suggestions
And each suggestion should have a confidence score
And suggestions should be ranked by confidence
And I should receive a flag indicating "ambiguous_classification": true

Scenario: Invalid input handling
Given empty or malformed business description
When I call the classification API
Then I should receive a 400 error
And error message should specify required fields
And error message should provide example of valid input
```

Implementation Requirements:

python

```
# API Request Schema
{
  "business_description": str,  # Required, 10-1000 characters
  "business_name": str,         # Optional, additional context
  "website_url": str,           # Optional, for website analysis
  "products_services": list,    # Optional, list of offerings
  "target_customers": str,      # Optional, B2B vs B2C context
  "country": str,               # Required, ISO country code
  "include_similar": bool       # Optional, include similar code suggestions
}

# API Response Schema
{
  "classification_id": str,       # Unique identifier
  "primary_classifications": {
    "mcc": {
      "code": str,
      "description": str,
      "confidence": float
    },
    "naics": {
      "code": str,
      "description": str,
      "confidence": float
    },
    "sic": {
      "code": str,
      "description": str,
      "confidence": float
    }
  },
  "alternative_suggestions": [
    {
```

```
            "code_type": str,
            "code": str,
            "description": str,
            "confidence": float,
            "similarity_score": float
        }
    ],
    "analysis_details": {
        "processing_time_ms": int,
        "model_version": str,
        "confidence_factors": list,
        "ambiguous_classification": bool
    },
    "timestamp": str,
    "expires_at": str
}
```

Story 1.2: Batch Classification Processing

As a payment processor with existing merchant portfolio
I want to classify multiple businesses in a single API call
So that I can efficiently process my entire merchant database

Acceptance Criteria:

```gherkin
```

```gherkin
Given I have a list of up to 1000 businesses to classify
When I submit a batch classification request
Then I should receive a job ID immediately
And I can check the job status using the job ID
And I receive webhook notifications when the job completes
And the results include individual classifications for each business
And failed classifications include error details
And the batch processing completes within 10 minutes for 1000 businesses

Scenario: Successful batch processing
Given a batch of 100 valid business records
When I submit to /api/v1/classify/batch
Then I should receive HTTP 202 with job_id
And I can GET /api/v1/classify/batch/{job_id} for status
And when complete, results include 100 successful classifications
And webhook is sent to configured endpoint

Scenario: Partial batch failure
Given a batch with 90 valid and 10 invalid records
When I submit the batch
Then valid records should be processed successfully
And invalid records should be marked with error details
And the job should complete with "partial_success" status
```

Story 1.3: Website-Based Classification Enhancement

As a risk analyst
I want the system to analyze merchant websites for more accurate classification
So that I get better context about the actual business operations

Acceptance Criteria:

```gherkin
Given a business with a valid website URL
When I request classification with website analysis enabled
Then the system should scrape and analyze the website content
And incorporate website findings into the classification
And provide website analysis details in the response
And handle websites that are unavailable or restricted
And respect robots.txt and rate limiting

Scenario: Website enhances classification accuracy
Given business description "Technology services" and website selling software
When website analysis is performed
Then classification should be more specific (e.g., "Software Publishers")
And response should indicate website analysis was used
And website_analysis section should show key findings

Scenario: Website analysis fails gracefully
Given a business with an inaccessible website
When website analysis is attempted
Then classification should proceed with description only
And response should indicate website analysis failed
And reason for failure should be provided
```

2.3 Technical Implementation Details

ML Model Architecture:

```python
```

```python
class BusinessClassificationPipeline:
    """
    End-to-end business classification pipeline
    """

    def __init__(self):
        self.text_preprocessor = BusinessTextPreprocessor()
        self.bert_classifier = BERTBusinessClassifier()
        self.similarity_matcher = SimilarityMatcher()
        self.confidence_calibrator = ConfidenceCalibrator()
        self.code_database = IndustryCodeDatabase()

    async def classify_business(self, business_data: dict) -> dict:
        """
        Main classification workflow
        """
        # 1. Preprocess and clean input text
        processed_text = await self.text_preprocessor.process(
            description=business_data.get('business_description'),
            name=business_data.get('business_name'),
            products=business_data.get('products_services', [])
        )

        # 2. Primary classification using BERT
        bert_predictions = await self.bert_classifier.predict(processed_text)

        # 3. Similarity-based backup classification
        similarity_predictions = await self.similarity_matcher.find_similar(
            processed_text, top_k=5
        )

        # 4. Ensemble predictions with confidence calibration
        final_predictions = await self.ensemble_predictions(
```

```python
        bert_predictions, similarity_predictions
    )

    # 5. Calibrate confidence scores
    calibrated_predictions = await self.confidence_calibrator.calibrate(
        final_predictions, processed_text
    )

    # 6. Generate response with alternatives
    return await self.format_response(calibrated_predictions, business_data)

async def ensemble_predictions(self, bert_preds, similarity_preds):
    """
    Combine BERT and similarity predictions using weighted ensemble
    """
    ensemble_weights = {
        'bert': 0.7,
        'similarity': 0.3
    }

    combined_scores = {}

    # Combine predictions for each code type
    for code_type in ['mcc', 'naics', 'sic']:
        bert_scores = bert_preds.get(code_type, {})
        sim_scores = similarity_preds.get(code_type, {})

        # Weighted combination
        for code in set(bert_scores.keys()) | set(sim_scores.keys()):
            bert_score = bert_scores.get(code, 0.0)
            sim_score = sim_scores.get(code, 0.0)

            combined_score = (
```

```
            bert_score * ensemble_weights['bert'] +
            sim_score * ensemble_weights['similarity']
        )

        if combined_score > 0.1:  # Minimum threshold
            combined_scores.setdefault(code_type, {})[code] = combined_score

    return combined_scores

class BusinessTextPreprocessor:
    """
    Preprocess business text for classification
    """

    def __init__(self):
        self.stop_words = self.load_industry_stop_words()
        self.business_synonyms = self.load_business_synonyms()

    async def process(self, description: str, name: str = None,
                     products: list = None) -> str:
        """
        Clean and preprocess business text
        """
        # Combine all available text
        text_parts = [description]
        if name:
            text_parts.append(name)
        if products:
            text_parts.extend(products)

        combined_text = " ".join(text_parts)

        # Text cleaning pipeline
```

```python
        cleaned_text = self.clean_text(combined_text)
        normalized_text = self.normalize_business_terms(cleaned_text)
        filtered_text = self.remove_noise(normalized_text)

        return filtered_text

    def clean_text(self, text: str) -> str:
        """Basic text cleaning"""
        import re

        # Remove special characters, keep alphanumeric and spaces
        text = re.sub(r'[^a-zA-Z0-9\s]', ' ', text)

        # Remove extra whitespace
        text = re.sub(r'\s+', ' ', text)

        # Convert to lowercase
        text = text.lower().strip()

        return text

    def normalize_business_terms(self, text: str) -> str:
        """Normalize business terminology"""
        # Replace synonyms with standard terms
        for synonym, standard in self.business_synonyms.items():
            text = text.replace(synonym, standard)

        return text
```

Code Database Schema:

```
sql
```

```sql
-- Industry codes lookup tables
CREATE TABLE mcc_codes (
    code VARCHAR(4) PRIMARY KEY,
    description TEXT NOT NULL,
    category VARCHAR(100),
    risk_level VARCHAR(20) DEFAULT 'medium',
    prohibited_countries TEXT[], -- JSON array of country codes
    requires_license BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE naics_codes (
    code VARCHAR(6) PRIMARY KEY,
    description TEXT NOT NULL,
    sector VARCHAR(2),
    sector_description TEXT,
    subsector VARCHAR(3),
    industry_group VARCHAR(4),
    naics_industry VARCHAR(5),
    level INTEGER, -- 2-digit, 3-digit, 4-digit, 5-digit, 6-digit
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE sic_codes (
    code VARCHAR(4) PRIMARY KEY,
    description TEXT NOT NULL,
    major_group VARCHAR(2),
    division_code VARCHAR(1),
    division_description TEXT,
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW()
```

```sql
);

-- Cross-reference mapping between code systems
CREATE TABLE code_mappings (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    mcc_code VARCHAR(4) REFERENCES mcc_codes(code),
    naics_code VARCHAR(6) REFERENCES naics_codes(code),
    sic_code VARCHAR(4) REFERENCES sic_codes(code),
    mapping_confidence DECIMAL(3,2), -- 0.00 to 1.00
    mapping_source VARCHAR(50), -- 'official', 'derived', 'ml_generated'
    created_at TIMESTAMP DEFAULT NOW()
);

-- Business synonym and keyword mappings
CREATE TABLE business_keywords (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    keyword VARCHAR(100) NOT NULL,
    code_type VARCHAR(10) NOT NULL, -- 'mcc', 'naics', 'sic'
    code VARCHAR(6) NOT NULL,
    weight DECIMAL(4,3) DEFAULT 1.000, -- Keyword importance weight
    context VARCHAR(50), -- 'primary', 'secondary', 'related'
    created_at TIMESTAMP DEFAULT NOW(),

    INDEX idx_keywords_lookup (keyword, code_type),
    INDEX idx_keywords_code (code_type, code)
);
```

---

3. Epic 2: Risk Assessment System

3.1 Epic Overview

Epic Description: Comprehensive risk assessment system that evaluates businesses across multiple risk dimensions and provides predictive risk scores with confidence intervals.

Business Value: Enables automated risk-based decision making, reduces manual review workload by 80%, and provides predictive insights to prevent future losses.

Success Metrics:

- Risk prediction accuracy: ≥85% for 6-month horizon
- Processing time: <3 seconds for comprehensive assessment
- Risk factor coverage: 50+ individual risk indicators
- Predictive capability: 3, 6, and 12-month risk forecasts

3.2 User Stories

Story 2.1: Real-time Risk Assessment

As a underwriting manager
I want to get instant risk scores for new merchant applications
So that I can make quick approval/rejection decisions

Acceptance Criteria:

```gherkin
```

```gherkin
Given a business with complete profile information
When I request a risk assessment
Then I should receive a comprehensive risk score within 3 seconds
And the score should be on a 1-100 scale (1=lowest risk, 100=highest risk)
And the response should include risk level classification (Low/Medium/High/Critical)
And individual risk category scores should be provided
And key risk factors should be identified and explained
And recommendations should be provided for risk mitigation

Scenario: Low risk business assessment
Given a well-established business with good web presence
When risk assessment is performed
Then overall score should be 1-25
And risk level should be "Low"
And positive risk factors should be highlighted
And minimal recommendations should be provided

Scenario: High risk business assessment
Given a newly registered business in high-risk industry
When risk assessment is performed
Then overall score should be 70-100
And risk level should be "High" or "Critical"
And specific risk factors should be detailed
And actionable mitigation recommendations should be provided

Scenario: Insufficient data handling
Given a business with minimal information available
When risk assessment is performed
Then assessment should complete with available data
And confidence interval should reflect data limitations
And recommendations should include data collection suggestions
```

API Specification:

```python
```

```
# Risk Assessment Request Schema
{
  "business_id": str,            # Required
  "assessment_type": str,        # "initial", "periodic", "triggered"
  "include_predictions": bool,    # Include 3/6/12 month forecasts
  "include_explanations": bool,   # Include risk factor explanations
  "risk_tolerance": str,         # "conservative", "moderate", "aggressive"
  "custom_weights": dict         # Optional custom risk category weights
}

# Risk Assessment Response Schema
{
  "assessment_id": str,
  "business_id": str,
  "overall_score": int,          # 1-100 scale
  "risk_level": str,            # "Low", "Medium", "High", "Critical"
  "confidence_interval": {
    "lower": float,            # Lower bound of confidence interval
    "upper": float,            # Upper bound of confidence interval
    "confidence_level": float  # e.g., 0.95 for 95% confidence
  },

  "risk_categories": {
    "operational_risk": {
      "score": int,
      "weight": float,
      "factors": [
        {
          "factor": str,
          "impact": str,      # "positive", "negative", "neutral"
          "severity": str,    # "low", "medium", "high"
          "explanation": str
        }
```

```
        ]
    },
    "financial_risk": {...},
    "regulatory_risk": {...},
    "reputational_risk": {...},
    "cybersecurity_risk": {...}
},

"predictions": {
    "3_month": {
        "predicted_score": int,
        "confidence": float,
        "trend": str,          # "increasing", "stable", "decreasing"
        "key_drivers": list
    },
    "6_month": {...},
    "12_month": {...}
},

"recommendations": [
    {
        "category": str,
        "priority": str,       # "high", "medium", "low"
        "action": str,
        "expected_impact": str,
        "timeline": str
    }
],

"data_quality": {
    "completeness_score": float,    # 0.0-1.0
    "freshness_score": float,       # 0.0-1.0
    "reliability_score": float,     # 0.0-1.0
```

```
        "missing_data_points": list
    },

    "model_metadata": {
        "model_version": str,
        "processing_time_ms": int,
        "data_sources_used": list,
        "last_model_update": str
    },

    "assessed_at": str,
    "valid_until": str
}
```

Story 2.2: Predictive Risk Modeling

As a portfolio risk manager
I want to see predicted risk evolution over time
So that I can proactively manage portfolio risk and prevent losses

Acceptance Criteria:

```gherkin
```

```
Given a business with historical data
When I request predictive risk assessment
Then I should receive 3, 6, and 12-month risk predictions
And each prediction should include confidence intervals
And trend analysis should indicate if risk is increasing/decreasing/stable
And key risk drivers for each time horizon should be identified
And early warning indicators should be highlighted

Scenario: Deteriorating risk trend
Given a business with declining key metrics
When predictive assessment is performed
Then 6-month prediction should show higher risk score
And trend should be marked as "increasing"
And specific drivers of increased risk should be identified
And early intervention recommendations should be provided

Scenario: Improving risk profile
Given a business with improving operational metrics
When predictive assessment is performed
Then future risk scores should show improvement
And positive trend factors should be highlighted
And recommendations should focus on sustaining improvements
```

Story 2.3: Risk Factor Analysis and Explanation

As a compliance officer
I want detailed explanations of why a business received a specific risk score
So that I can document decisions and ensure regulatory compliance

Acceptance Criteria:

```gherkin
```

```
Given any risk assessment result
When I request detailed explanations
Then each risk factor should have a clear explanation
And the impact of each factor on the overall score should be quantified
And explanations should be in plain English, not technical jargon
And supporting evidence should be provided where possible
And explanations should be suitable for regulatory documentation

Scenario: High-risk classification explanation
Given a business classified as high-risk
When detailed explanations are provided
Then specific factors contributing to high risk should be listed
And each factor should include severity level and impact score
And regulatory implications should be noted where relevant
And mitigation strategies should be suggested for each factor
```

3.3 Risk Assessment Categories

Operational Risk Factors:

```python
```

```
OPERATIONAL_RISK_FACTORS = {
    'business_age': {
        'weight': 0.15,
        'calculation': lambda days: max(0, 50 - (days / 30)),  # Newer = higher risk
        'explanation': 'Newer businesses have higher operational uncertainty'
    },
    'business_registration_completeness': {
        'weight': 0.10,
        'factors': ['legal_name', 'tax_id', 'registration_date', 'registered_address'],
        'calculation': lambda missing: len(missing) * 10,
        'explanation': 'Incomplete registration indicates potential legitimacy issues'
    },
    'website_quality': {
        'weight': 0.12,
        'sub_factors': {
            'ssl_certificate': 0.3,
            'professional_design': 0.25,
            'contact_information': 0.25,
            'privacy_policy': 0.2
        },
        'explanation': 'Poor web presence suggests unprofessional operations'
    },
    'social_media_presence': {
        'weight': 0.08,
        'platforms': ['facebook', 'linkedin', 'twitter', 'instagram'],
        'calculation': 'calculate_social_presence_score',
        'explanation': 'Limited social presence may indicate fake or inactive business'
    },
    'customer_reviews': {
        'weight': 0.10,
        'sources': ['google', 'yelp', 'trustpilot', 'bbb'],
        'factors': ['review_count', 'average_rating', 'recent_activity'],
        'explanation': 'Poor customer feedback indicates service quality issues'
```

```
    },
    'contact_information_validity': {
        'weight': 0.08,
        'checks': ['phone_verification', 'address_verification', 'email_verification'],
        'explanation': 'Invalid contact information suggests potential fraud'
    },
    'industry_risk_profile': {
        'weight': 0.20,
        'risk_categories': {
            'adult_entertainment': 85,
            'gambling': 80,
            'cryptocurrency': 75,
            'travel_agencies': 65,
            'restaurants': 45,
            'retail_clothing': 25,
            'software_services': 20
        },
        'explanation': 'Some industries have inherently higher operational risks'
    },
    'seasonal_business_indicators': {
        'weight': 0.05,
        'calculation': 'detect_seasonality_patterns',
        'explanation': 'Highly seasonal businesses face cash flow challenges'
    }
}

FINANCIAL_RISK_FACTORS = {
    'estimated_revenue_stability': {
        'weight': 0.25,
        'indicators': ['revenue_growth_rate', 'revenue_consistency', 'market_trends'],
        'explanation': 'Unstable revenue indicates financial distress risk'
    },
    'payment_processing_history': {
```

```
        'weight': 0.20,
        'factors': ['processing_length', 'chargeback_ratio', 'refund_ratio'],
        'thresholds': {'chargeback_ratio': 0.02, 'refund_ratio': 0.10},
        'explanation': 'Poor payment history indicates customer dissatisfaction'
    },
    'credit_indicators': {
        'weight': 0.15,
        'sources': ['business_credit_score', 'trade_references', 'bank_relationships'],
        'explanation': 'Poor credit history suggests financial management issues'
    },
    'cash_flow_indicators': {
        'weight': 0.15,
        'proxies': ['payment_terms', 'inventory_turnover', 'accounts_receivable'],
        'explanation': 'Poor cash flow management increases failure risk'
    },
    'debt_to_equity_estimates': {
        'weight': 0.10,
        'calculation': 'estimate_leverage_ratio',
        'explanation': 'High leverage increases financial distress probability'
    },
    'market_competition': {
        'weight': 0.10,
        'factors': ['market_saturation', 'competitive_advantages', 'barriers_to_entry'],
        'explanation': 'Intense competition affects profitability and survival'
    },
    'economic_sensitivity': {
        'weight': 0.05,
        'indicators': ['economic_cycle_correlation', 'discretionary_spending_exposure'],
        'explanation': 'Economic downturns disproportionately affect some businesses'
    }
}

REGULATORY_RISK_FACTORS = {
```

```
    'sanctions_screening_results': {
        'weight': 0.30,
        'lists': ['ofac_sdn', 'un_sanctions', 'eu_sanctions'],
        'match_types': ['exact', 'close', 'possible'],
        'explanation': 'Sanctions matches indicate legal and compliance risks'
    },
    'license_requirements': {
        'weight': 0.20,
        'checks': ['required_licenses', 'license_status', 'license_expiration'],
        'explanation': 'Operating without required licenses creates legal liability'
    },
    'regulatory_violations_history': {
        'weight': 0.15,
        'sources': ['sec_filings', 'ftc_actions', 'state_regulators'],
        'explanation': 'Past violations suggest ongoing compliance issues'
    },
    'data_protection_compliance': {
        'weight': 0.10,
        'frameworks': ['gdpr', 'ccpa', 'hipaa'],
        'indicators': ['privacy_policy', 'data_handling_practices'],
        'explanation': 'Data breaches create significant regulatory exposure'
    },
    'anti_money_laundering_risk': {
        'weight': 0.15,
        'factors': ['cash_intensive_business', 'high_risk_geography', 'complex_ownership'],
        'explanation': 'AML violations carry severe regulatory penalties'
    },
    'tax_compliance_indicators': {
        'weight': 0.10,
        'checks': ['tax_id_validity', 'tax_lien_searches', 'compliance_history'],
        'explanation': 'Tax issues indicate potential business instability'
    }
}
```

This completes Part 1 of the Feature Specifications Document, covering the core Business Classification Engine and Risk Assessment System with detailed user stories, acceptance criteria, and technical implementation details.

Should I continue with **Part 2: Web Dashboard, API Specifications, and Compliance Features**?