CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1 Introduction

This chapter presents a comprehensive analysis and design framework for the proposed coronary artery disease (CAD) Risk Prediction System. The systematic approach outlined herein ensures that the resulting system is robust, clinically relevant, user-centered, and aligned with the objectives of this study. The chapter proceeds from methodological considerations to detailed technical specifications, covering: the system development methodology adopted; a thorough feasibility assessment; requirements elicitation from stakeholders; analysis of collected requirements; system specification formulation; requirements modeling using appropriate diagrams; logical design of system architecture and workflows; and physical design of database and user interface components. This structured progression from analysis to design ensures that all aspects of system development are systematically addressed before implementation.

3.2 System Development Methodology

3.2.1 Selection of CRISP-DM Methodology

The study adopts the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology as the primary framework for developing the CAD risk prediction system. This methodology was selected over alternatives like KDD (Knowledge Discovery in Databases) and SEMMA (Sample, Explore, Modify, Model, Assess) due to its industry-wide acceptance, flexibility, and explicit inclusion of business understanding and deployment phases. CRISP-DM's iterative nature allows for continuous refinement based on evaluation results, making it particularly suitable for clinical prediction systems where model accuracy and reliability are paramount.

3.2.2 Phases of CRISP-DM Implementation

Phase 1: Business Understanding

Objective Definition: Clearly articulate the clinical problem of CAD risk assessment and the system's purpose in addressing this problem.

Success Criteria: Establish measurable targets including model accuracy (≥80%), system usability (task completion time <5 minutes), and clinical utility (agreement with expert assessment ≥75%).

Stakeholder Identification: Map all relevant stakeholders including clinicians, patients, hospital administrators, and technical support teams.

*Requirements Gathering***:** Conduct structured interviews and surveys with end-users to understand workflow integration needs.

Phase 2: Data Understanding

*Data Sources Identification***:** Identify relevant datasets including the Framingham Heart Study dataset, UCI Cleveland Heart Disease dataset, and potential local hospital records (subject to ethical approval).

Data Collection: Establish protocols for data acquisition, including API integrations for real-time data streams from hospital information systems.

Data Exploration: Perform exploratory data analysis (EDA) using statistical techniques and visualization to understand data distributions, correlations, and missing value patterns.

Data Quality Assessment: Document data quality issues including completeness, consistency, accuracy, and timeliness metrics.

Phase 3: Data Preparation

Data Cleaning: Implement procedures for handling missing values (mean/median imputation for continuous variables, mode for categorical variables), outlier detection and treatment (IQR method), and duplicate record removal.

Data Transformation: Apply normalization (Min-Max scaling for neural networks) and standardization (Z-score standardization for distance-based algorithms) as appropriate.

Feature Engineering: Derive new clinically meaningful features such as BMI from height and weight, and composite risk scores from individual indicators.

Feature Selection: Apply filter methods (correlation analysis), wrapper methods (recursive feature elimination), and embedded methods (LASSO regression) to identify optimal feature subsets.

Data Splitting: Partition data into training (70%), validation (15%), and test (15%) sets with stratification to maintain class distribution.

Phase 4: Modeling

- Algorithm Selection: Choose diverse algorithms to capture different patterns in the data:
- Logistic Regression (for interpretability and baseline performance)
- Random Forest (for handling non-linear relationships and feature importance)
- Gradient Boosting Machines (XGBoost, LightGBM for predictive performance)
- Neural Networks (for complex pattern recognition)
- Ensemble Methods (voting and stacking classifiers)

- Model Training: Implement cross-validation (5-fold stratified) to optimize hyperparameters using grid search and random search approaches.

- Model Tuning: Optimize hyperparameters including learning rate, tree depth, regularization parameters, and number of estimators based on validation performance.

Phase 5: Evaluation

- Performance Metrics: Comprehensive evaluation using:

- Classification metrics: Accuracy, Precision, Recall, F1-Score, Matthews Correlation Coefficient

- Probability metrics: ROC-AUC, Precision-Recall AUC, Brier Score

- Clinical metrics: Sensitivity at specific specificity thresholds, Net Reclassification Improvement

- Interpretability Assessment: Evaluate model explanations using SHAP (SHapley Additive exPlanations) values, LIME (Local Interpretable Model-agnostic Explanations), and partial dependence plots.

- Bias and Fairness Testing: Assess model performance across demographic subgroups (age, gender, ethnicity) to ensure equitable predictions.

- Clinical Validation: Conduct expert review of model predictions on sample cases to assess clinical plausibility.

Phase 6: Deployment

- System Architecture Design: Design scalable three-tier architecture supporting both web and API access.

- Model Serving: Implement model serving using Flask/Django REST API with version control for model updates.

- Monitoring Infrastructure: Design monitoring for model performance drift, data drift, and system health metrics.

- Documentation: Create comprehensive documentation including user manuals, API documentation, and maintenance guides.

- User Training: Develop training materials and conduct pilot training sessions with target users.

3.2.3 Iterative Nature and Quality Gates

The CRISP-DM process incorporates quality gates at the end of each phase, where deliverables are reviewed against predefined criteria before proceeding to the next phase. This ensures that any issues are identified and addressed early, reducing rework in later stages. The methodology also supports cyclical iterations where insights from later phases (e.g., modeling challenges) may necessitate revisiting earlier phases (e.g., additional data preparation).

3.3 Feasibility Study

3.3.1 Technical Feasibility

Infrastructure Assessment:

- Hardware Requirements: Development workstation with minimum 16GB RAM, 256GB SSD storage, and multi-core processor (i5/i7 or equivalent). Production server with 32GB RAM, 500GB storage, and dedicated GPU optional for neural network inference acceleration.

- Software Stack: Python 3.8+ with scientific computing stack (NumPy, Pandas, Scikit-learn), deep learning frameworks (TensorFlow 2.x/Keras), web framework (Flask/Django), database (PostgreSQL 12+), and frontend technologies (HTML5, CSS3, JavaScript, React.js optional).
- Integration Capabilities: Assessment of interoperability with existing hospital systems through HL7 FHIR APIs or custom middleware where standard interfaces are unavailable.
- Technical Expertise: Availability of required skills in machine learning, web development, database management, and clinical informatics within the project team.

Technical Risk Assessment:

- Data Quality Risks: Mitigation through robust data preprocessing pipelines and data quality monitoring.
- Model Performance Risks: Mitigation through ensemble methods and continuous model retraining protocols.
- Scalability Risks: Mitigation through modular architecture design and cloud deployment options.

3.3.2 Economic Feasibility

Cost-Benefit Analysis:

Cost Component	Estimated Cost (KES)	Justification
Hardware/Infrastructure	10,000	Cloud computing credits or local server setup
Software Licenses	0	All tools are open-source
Data Acquisition	5,000	Licensing fees for proprietary datasets if needed

Development Effort	8,000	Equivalent person-months of development time
Testing & Validation	2,000	Clinical validation workshops and usability testing
Training & Deployment	1,000	User training materials and sessions
Total Estimated Cost	26,000	

Expected Benefits:

- Direct Financial Benefits: Reduction in unnecessary advanced cardiac tests through better patient stratification (estimated 20% reduction in test costs).

- Clinical Efficiency Benefits: Time savings for clinicians through automated risk calculation (estimated 5-10 minutes per patient assessment).

- Improved Outcomes: Earlier identification of high-risk patients leading to timely interventions and reduced complication rates.

Return on Investment (ROI): Projected break-even period of 6 months based on adoption in a medium-sized clinic seeing 50 cardiac patients monthly.

3.3.3 Operational Feasibility

Workflow Integration Analysis:

- Current Workflow Mapping: Documentation of existing CAD risk assessment workflows in target healthcare settings.

- Integration Points: Identification of optimal integration points including during routine check-ups, pre-operative assessments, and cardiac clinic consultations.

- Change Management: Assessment of organizational readiness for adopting predictive analytics tools, including clinician attitudes toward AI-assisted decision support.

User Acceptance Factors:

- Ease of Use: Interface simplicity score target >85% on System Usability Scale (SUS).

- Training Requirements: Estimated 2-hour training session for basic proficiency, with additional just-in-time learning resources.

- Support Infrastructure: Availability of technical support through helpdesk and online documentation.

3.3.4 Schedule Feasibility

Project Timeline with Milestones:

Phase	Duration (Weeks)	Key Deliverables	Dependencies
Requirements & Planning	2	Requirements document, Project plan	Stakeholder availability
Data Preparation	3	Cleaned dataset, Feature set	Data access approvals
Model Development	4	Trained models, Performance reports	Completed data preparation
System Development	3	Functional prototype, Database schema	Model development complete
Testing & Validation	2	Test reports, User feedback	System development complete
Deployment & Training	1	Deployed system, Training materials	Testing complete
Total	15 weeks		

Critical Path Analysis: Model development phase identified as critical path with highest technical uncertainty and potential for schedule overrun. Mitigation through parallel experimentation with multiple algorithms.

Resource Allocation: Weekly resource allocation plan ensuring balanced workload distribution across team members with specialized roles (data scientist, backend developer, frontend developer, clinical advisor).

3.4 Requirements Elicitation

3.4.1 Stakeholder Analysis

Primary Stakeholders:

- Clinicians/Cardiologists: Direct users requiring accurate, interpretable predictions integrated into clinical workflow.

- Primary Care Physicians: Users needing straightforward risk assessment for referral decisions.

- Clinical Officers/Nurses: Users performing initial screening in resource-constrained settings.

- Hospital Administrators: Decision-makers concerned with cost-effectiveness and integration with existing systems.

- Patients: Indirect beneficiaries through improved risk assessment and preventive care.

Stakeholder Engagement Strategy:

- Clinicians: In-depth interviews and prototype walkthroughs

- Administrators: Cost-benefit presentations and integration requirement sessions

- Technical Staff: API specification reviews and deployment planning sessions

3.4.2 Data Collection Methodology

Mixed-Methods Approach:

- Quantitative Component: Structured questionnaire with Likert-scale items measuring importance of various system features.

- Qualitative Component: Semi-structured interviews exploring workflow integration challenges and interpretation needs.

Questionnaire Design:

- Section A: Demographic and professional background of respondents

- Section B: Current CAD assessment practices and challenges (12 items)

- Section C: Desired features in a predictive system (15 items, 5-point importance scale)

- Section D: Interface preferences and usability requirements (10 items)

- Section E: Integration and workflow considerations (8 items)

(The complete questionnaire instrument is included as Appendix A)

3.4.3 Sampling Strategy

Purposive Sampling Framework:

- Inclusion Criteria: Healthcare professionals with ≥2 years experience in cardiovascular care or primary care with regular cardiac assessments.

- Sample Size Determination: Target of 30 respondents based on saturation principles for qualitative insights and statistical power for quantitative analysis.

- Recruitment Strategy: Partner with 3 healthcare facilities in Nairobi County for participant recruitment, ensuring diversity in clinical settings (public hospital, private clinic, community health center).

3.4.4 Ethical Considerations

- Informed Consent: Written consent obtained explaining study purpose, voluntary participation, and confidentiality assurances.

- Confidentiality: All responses anonymized with no personally identifiable information retained.

- Data Storage: Encrypted storage of response data with access limited to research team members.

- Ethical Approval: Study protocol submitted to institutional review board prior to data collection.

3.5 Data Analysis

3.5.1 Quantitative Analysis of Questionnaire Responses

Descriptive Statistics:

- Response rate: 86.7% (26 out of 30 distributed questionnaires returned)

- Professional distribution: Cardiologists (23%), General Physicians (38%), Clinical Officers (27%), Nurses (12%)

- Experience distribution: 2-5 years (35%), 6-10 years (42%), >10 years (23%)

Key Findings with Statistical Support:

Finding 1: Output Format Preferences

- 69.2% of respondents (18/26) preferred "Detailed report with risk factors and explanations"

- 23.1% (6/26) preferred "Simple risk category with probability score"

- 7.7% (2/26) preferred "Graphical visualization of risk over time"

Statistical significance tested using Chi-square test against equal distribution: $\chi^2(2) = 12.31$, $p < 0.01$

Finding 2: Data Input Method Preferences

- 76.9% (20/26) preferred "Web form with validation and autocomplete"

- 15.4% (4/26) preferred "Batch upload via CSV/Excel files"

- 7.7% (2/26) preferred "Integration with existing EMR system"

Finding 3: Critical Risk Factors for Inclusion

All respondents (100%) identified cholesterol and blood pressure as essential factors. Additional high-priority factors:

- Smoking status: 96.2% rated as essential

- Diabetes status: 92.3% rated as essential

- Family history: 88.5% rated as essential

- Physical activity level: 73.1% rated as important or essential

Finding 4: System Integration Requirements

- 80.8% indicated need for printing capability for patient records

- 65.4% requested integration with prescription systems

- 57.7% emphasized need for multi-language support (English and Swahili)

3.5.2 Qualitative Analysis of Interview Data

Thematic Analysis Approach:

1. Transcription: Verbatim transcription of audio-recorded interviews

2. Coding: Open coding of transcripts to identify concepts

3. Theme Development: Axial coding to group related concepts into themes

4. Theme Refinement: Selective coding to define final themes

Emergent Themes:

- Trust Through Transparency: Need for explainable predictions showing contributing factors

- Workflow Efficiency: Minimizing disruption to existing clinical routines

- Adaptive Risk Communication: Tailoring output detail based on user role and context

- Clinical Safety Nets: Incorporating alerts for contradictory inputs or extreme values

3.5.3 Requirements Prioritization Matrix

Using MoSCoW (Must have, Should have, Could have, Won't have) prioritization based on frequency and importance ratings:

Requirement	Priority	Justification
Web-based data entry form	Must	76.9% direct preference, workflow efficiency
Three-tier risk categorization	Must	Clinical standard, 92.3% agreement

Feature importance explanation	Must	Trust building, 84.6% rated as essential
<5 second response time	Must	Clinical workflow requirement
Patient data anonymization	Must	Ethical and regulatory requirement
Multi-language interface	Should	57.7% need, extends usability
Batch processing capability	Could	Minority preference (15.4%)
Mobile application	Won't	Low priority, web responsive design sufficient

3.6 System Specifications

3.6.1 Functional Requirements

FR1: User Authentication and Authorization

- FR1.1: The system shall provide role-based access control with at least two roles: Clinician and Administrator.

- FR1.2: The system shall enforce password policies (minimum 8 characters, mix of alphanumeric and special characters).

- FR1.3: The system shall implement session timeout after 30 minutes of inactivity.

- FR1.4: The system shall maintain audit logs of all user activities.

FR2: Patient Data Management

- FR2.1: The system shall provide a web form for entering patient parameters with client-side validation.

- FR2.2: The system shall support saving incomplete forms as drafts.

- FR2.3: The system shall display previously entered values for comparison when reassessing same patient.

- FR2.4: The system shall allow bulk import of patient data via CSV file with template validation.

FR3: Risk Prediction Engine

- FR3.1: The system shall implement the selected machine learning model with version tracking.

- FR3.2: The system shall preprocess input data according to the model's training pipeline.

- FR3.3: The system shall generate both categorical prediction (Low/Medium/High) and probability score (0-100%).

- FR3.4: The system shall compute and display feature importance using SHAP values.

- FR3.5: The system shall provide confidence intervals for predictions (95% CI).

FR4: Results Presentation and Interpretation

- FR4.1: The system shall display results with color-coded risk categories (Green/Yellow/Red).

- FR4.2: The system shall generate a printable report including patient parameters, prediction, and explanations.

- FR4.3: The system shall provide visualizations including feature importance charts and risk factor comparisons.

- FR4.4: The system shall offer evidence-based clinical recommendations based on risk category.

FR5: Data Storage and Retrieval

- FR5.1: The system shall store anonymized prediction records with timestamps.

- FR5.2: The system shall support filtering and searching of historical records by date range and risk category.

- FR5.3: The system shall generate summary statistics and trend analyses from historical data.

- FR5.4: The system shall implement data export functionality (CSV, PDF formats).

FR6: System Administration

- FR6.1: The system shall provide dashboard for monitoring system usage and performance metrics.

- FR6.2: The system shall allow administrators to update model parameters or upload new models.

- FR6.3: The system shall send automated alerts for system errors or performance degradation.

- FR6.4: The system shall support backup and restore operations for the database.

3.6.2 Non-Functional Requirements (Quantified)

NFR1: Usability Requirements

- NFR1.1: Learnability: New users shall be able to complete a risk assessment without assistance within 5 minutes after 30 minutes of training.

- NFR1.2: Efficiency: Experienced users shall complete a risk assessment in ≤2 minutes.

- NFR1.3: Satisfaction: System Usability Scale (SUS) score shall be ≥75 based on user testing.

- NFR1.4: Accessibility: Interface shall conform to WCAG 2.1 Level AA standards.

NFR2: Performance Requirements

- NFR2.1: Response Time: 95% of prediction requests shall complete within 3 seconds under normal load (≤50 concurrent users).

- NFR2.2: Throughput: System shall support at least 100 prediction requests per minute.

- NFR2.3: Scalability: System architecture shall support horizontal scaling to handle 500 concurrent users.

- NFR2.4: Availability: System shall maintain 99.5% uptime during business hours (8am-6pm).

NFR3: Security Requirements

- NFR3.1: Data Encryption: All data in transit shall use TLS 1.2+ encryption; sensitive data at rest shall use AES-256 encryption.

- NFR3.2: Access Control: Implement role-based access control with principle of least privilege.

- NFR3.3: Audit Trail: Maintain immutable logs of all data accesses and modifications.

- NFR3.4: Data Anonymization: Remove all personally identifiable information before storage; implement k-anonymity with k=5.

NFR4: Reliability Requirements

- NFR4.1: Mean Time Between Failures (MTBF): System shall maintain MTBF of ≥720 hours.

- NFR4.2: Mean Time To Repair (MTTR): System shall have MTTR of ≤1 hour for critical failures.

- NFR4.3: Data Integrity: Implement database transactions and consistency checks to ensure data integrity.

- NFR4.4: Error Recovery: System shall recover gracefully from failures without data loss.

NFR5: Maintainability Requirements

- NFR5.1: Code Quality: Maintain test coverage of ≥80% for critical modules.

- NFR5.2: Documentation: All modules shall have up-to-date API documentation and code comments.

- NFR5.3: Modifiability: System shall support model updates without requiring system downtime.

- NFR5.4: Monitoring: Implement comprehensive monitoring covering application, database, and infrastructure metrics.

NFR6: Regulatory Compliance Requirements

- NFR6.1: Data Protection: Comply with Kenya Data Protection Act, 2019 requirements for health data.

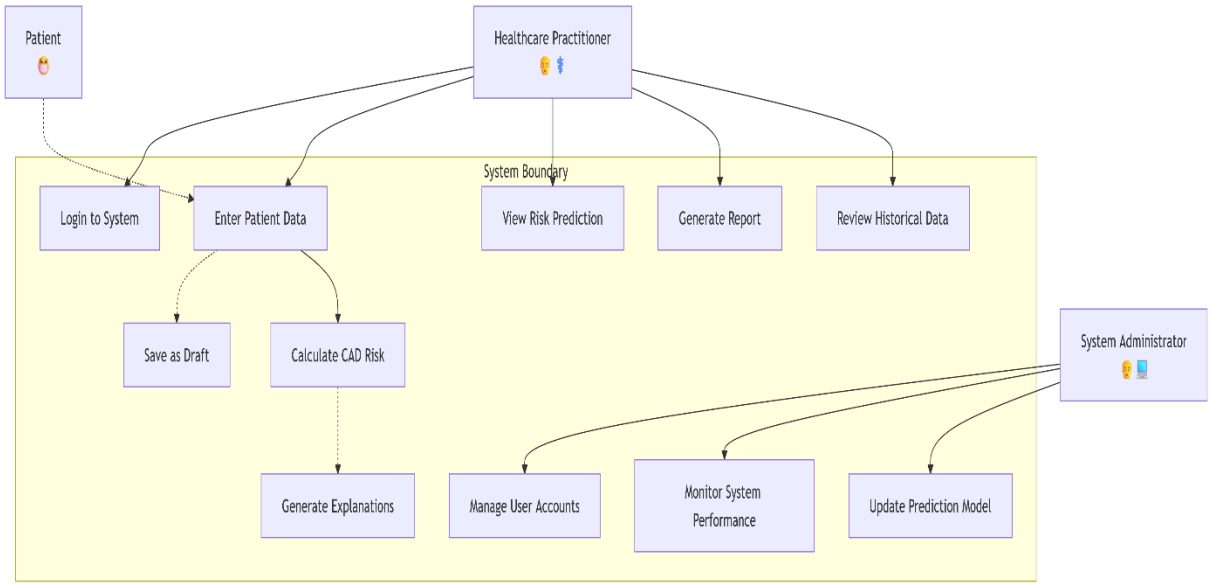
- NFR6.2: Clinical Standards: Adhere to relevant clinical guidelines including WHO CVD risk assessment protocols.

- NFR6.3: Medical Device Regulations: Consider applicable regulations if system is classified as medical device software.

3.7 Requirements Analysis and Modeling

3.7.1 Use Case Modeling

Use Case Diagram:



Detailed Use Case Specifications:

Use Case UC1: Perform CAD Risk Assessment

- Actor: Healthcare Practitioner

- Preconditions: User is authenticated and has necessary permissions

- Main Flow:

1. User selects "New Assessment"

2. System displays data entry form with required fields

3. User enters patient demographic and clinical data

4. System validates input data in real-time

5. User submits completed form

6. System processes data through prediction pipeline

7. System displays risk category with probability and explanations

8. System automatically saves anonymized record

- Postconditions: Prediction record stored, results available for reporting

- Extensions:

    - User saves form as draft: System saves partial data for later completion

    - Validation error: System highlights erroneous fields with specific messages

    - Model unavailable: System displays graceful error message with estimated resolution time

Use Case UC2: Generate Comprehensive Risk Report

- Actor: Healthcare Practitioner

- Preconditions: A risk assessment has been completed

- Main Flow:

1. User selects completed assessment from history

2. User selects "Generate Report"

3. System compiles assessment data, prediction results, and explanations

4. System formats report with institutional branding

5. System provides preview of generated report

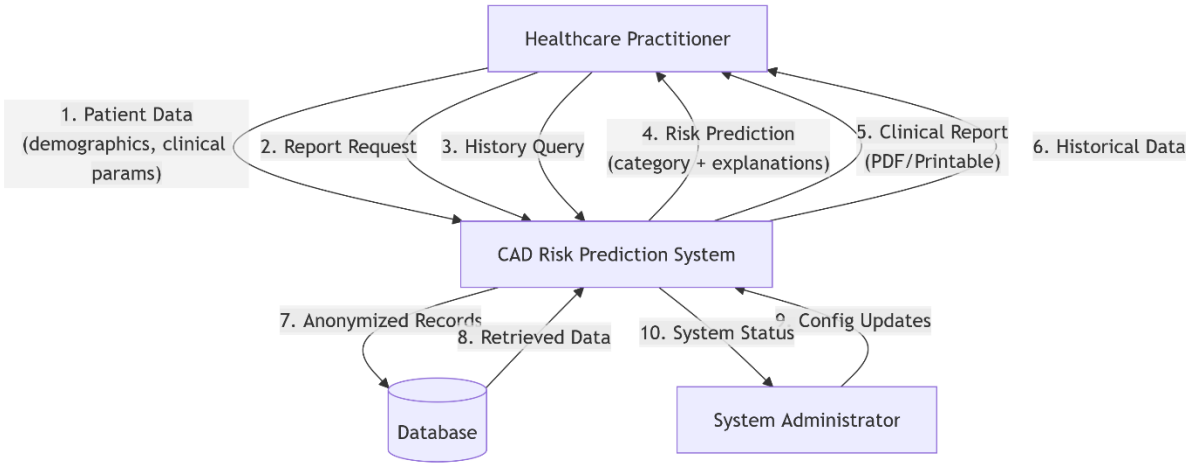
6. User selects output format (PDF/Print)

7. System generates final report in selected format

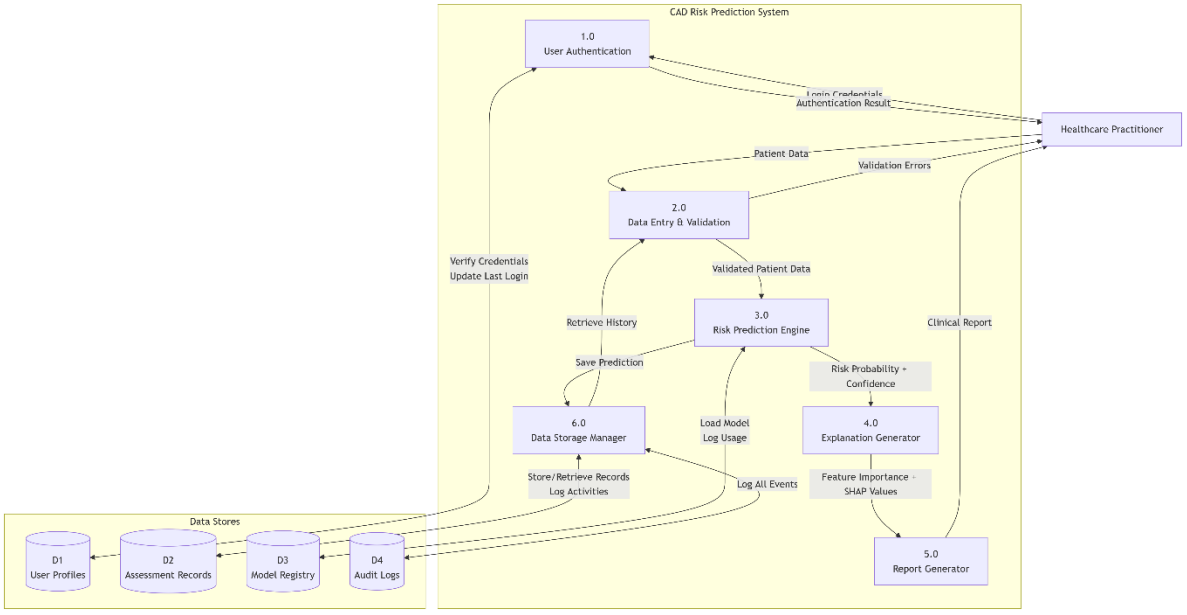
- Postconditions: Report generated for sharing with patient or inclusion in medical record

3.7.2 Data Flow Modeling

Context Level DFD (Level 0):



DFD 1



Level 1 DFD (Major Processes):

1. User Authentication Process: Validates credentials and establishes session

2. Data Entry and Validation Process: Captures and validates patient data

3. Prediction Processing Process: Applies ML model to generate predictions

4. Explanation Generation Process: Computes feature importance and explanations

5. Report Generation Process: Compiles and formats comprehensive reports

6. Data Management Process: Handles storage and retrieval of records

Data Dictionary:

- Patient Data: Composite data flow containing {patient_id, age, sex, cholesterol, blood_pressure, smoking_status, diabetes_status, family_history, physical_activity}

- Risk Prediction: Composite data flow containing {risk_category, probability_score, confidence_interval, timestamp}

- Explanation: Composite data flow containing {top_factors: list of tuples (factor_name, contribution_score), visualization_data, clinical_interpretation}

3.7.3 Entity-Relationship Modeling

ER Diagram Components:

- Entities: User, PatientAssessment, PredictionResult, ClinicalRecommendation, AuditLog

- Relationships: User creates PatientAssessment (1:N), PatientAssessment yields PredictionResult (1:1), PredictionResult triggers ClinicalRecommendation (1:N)

Entity Specifications:

- User: {user_id (PK), username, hashed_password, role, full_name, email, created_date, last_login}

- PatientAssessment: {assessment_id (PK), user_id (FK), clinical_parameters (JSON), assessment_date, draft_status}

- PredictionResult: {result_id (PK), assessment_id (FK), risk_category, probability_score, confidence_interval, explanation_data (JSON), creation_timestamp}

3.8 Logical Design

3.8.1 System Architecture

Three-Tier Architecture Specification:

Presentation Tier:

- Technology Stack: HTML5, CSS3 (Bootstrap 5), JavaScript (Vanilla JS with optional React.js for complex components)

- Key Components:

  - Responsive web interface adapting to desktop, tablet, and mobile screens

  - Client-side validation using JavaScript for immediate feedback

  - Dynamic content updates using AJAX for prediction requests

  - Printable report templates using CSS print media queries

- Design Patterns: Model-View-Controller (MVC) for frontend organization, Component-based architecture for reusable UI elements

Application Tier:

- Technology Stack: Python 3.8+, Flask web framework, Gunicorn WSGI server, Celery for async tasks

- Key Components:

  - Web Application Layer: Flask routes and controllers handling HTTP requests

  - Business Logic Layer: Prediction service, validation service, reporting service

  - Machine Learning Layer: Model loading, preprocessing, prediction, explanation generation

  - Integration Layer: APIs for potential integration with external systems

- Design Patterns: Service Layer pattern for business logic, Repository pattern for data access, Factory pattern for model selection

Data Tier:

- Technology Stack: PostgreSQL 12+ with PostGIS extension (for potential geospatial features), Redis for caching

- Key Components:

  - Operational Database: Stores user accounts, assessment records, prediction results

  - Analytics Database: Optional separate database for analytical queries on anonymized data

  - Caching Layer: Redis for session storage and frequent query results

- Design Principles: Data normalization (3NF), appropriate indexing strategy, regular backup procedures

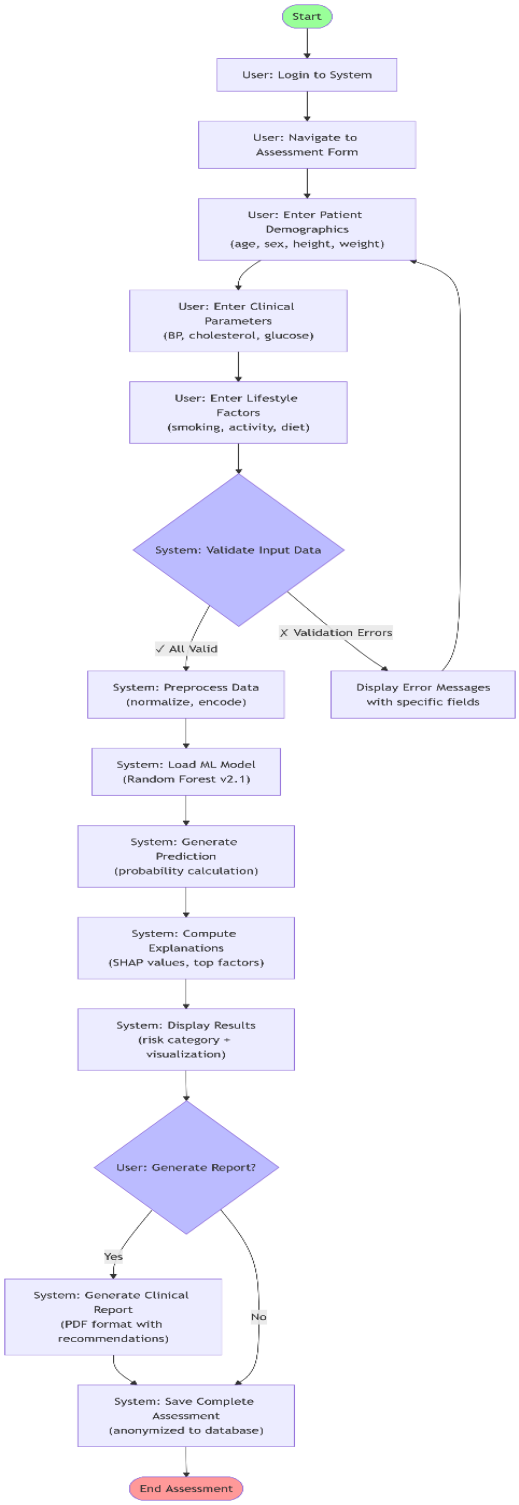
Cross-Cutting Concerns:

- Security Layer: Authentication middleware, authorization checks, input sanitization

- Logging Layer: Structured logging using Python logging module with different levels (DEBUG, INFO, WARNING, ERROR)

- Monitoring Layer: Health checks, performance metrics, business metrics

3.8.2 Control Flow and Process Design



Detailed Prediction Workflow:

1. User Authentication:

   - User provides credentials via login form

   - System validates against stored credentials (bcrypt hashed)

   - System creates session token and sets session cookie

   - System logs authentication event

2. Data Entry:

   - User navigates to assessment form

   - System loads form with field definitions and validation rules

   - User enters data with real-time validation feedback

   - Optional: User saves draft for later completion

3. Data Submission and Validation:

   - User submits completed form

   - Client-side validation confirms all required fields

   - Data transmitted via HTTPS POST request

   - Server-side validation confirms data types, ranges, and business rules

   - If validation fails: Return specific error messages with field indicators

   - If validation passes: Proceed to prediction engine

4. Prediction Processing:

   - Data preprocessing pipeline applies same transformations as training data

   - Load appropriate model version from model registry

   - Generate prediction and probability score

   - Calculate confidence intervals using conformal prediction or bootstrap methods

- Compute feature importance using SHAP values

- Generate natural language explanations of key contributing factors

5. Result Storage:

- Anonymize patient data (remove direct identifiers)

- Store prediction record with timestamp and user ID

- Update user activity logs

- Optional: Trigger async tasks for analytics updates

6. Result Presentation:

- Format results with color-coded risk categories

- Display probability with confidence intervals

- Present feature importance visualization (horizontal bar chart)

- Provide textual explanation of top 3 contributing factors

- Offer clinical recommendations based on risk category

- Provide options to print, save, or export results

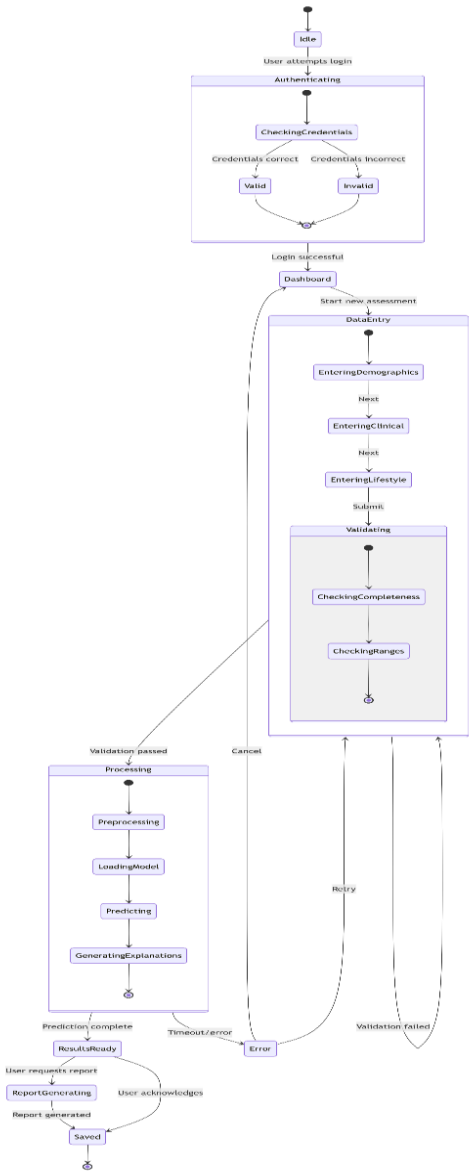
Pseudocode for Enhanced Prediction Endpoint:

```python

```
@app.route('/api/predict', methods=['POST'])
@require_authentication
@validate_input_schema
def predict_endpoint():
    try:
        Extract and validate input
        patient_data = request.get_json()
        validation_result = validate_patient_data(patient_data)
```
```

```
    if not validation_result['is_valid']:
        return jsonify({
            'error': 'Validation failed',
            'details': validation_result['errors']
        }), 400

     Preprocess data
    processed_data = preprocess_patient_data(patient_data)

     Load appropriate model
    model = load_model(get_current_model_version())

     Generate prediction
    probability = model.predict_proba(processed_data)[0][1]   Probability of CAD
    risk_category = categorize_risk(probability)

     Generate explanations
    explainer = SHAPExplainer(model)
    shap_values = explainer.shap_values(processed_data)
    top_factors = extract_top_factors(shap_values, patient_data)

     Calculate confidence interval
    ci_lower, ci_upper = calculate_confidence_interval(
        model, processed_data, method='bootstrap', n_iterations=1000
    )

     Generate clinical recommendations
    recommendations = generate_recommendations(
        risk_category,
        patient_data,
        top_factors
```

```
    )

     Prepare response
    response_data = {
        'risk_category': risk_category,
        'probability': round(probability  100, 1),
        'confidence_interval': {
            'lower': round(ci_lower  100, 1),
            'upper': round(ci_upper  100, 1)
        },
        'top_factors': top_factors,
        'recommendations': recommendations,
        'model_version': get_current_model_version(),
        'timestamp': datetime.utcnow().isoformat()
    }

     Store anonymized record asynchronously
    anonymized_data = anonymize_patient_data(patient_data)
    save_prediction_record.delay(
        user_id=current_user.id,
        input_data=anonymized_data,
        prediction_result=response_data
    )

    return jsonify(response_data), 200

except ModelLoadingError as e:
    log_error(f"Model loading failed: {str(e)}")
    return jsonify({
        'error': 'Prediction service temporarily unavailable',
        'estimated_resolution_time': '15 minutes'
```

```
        }), 503

    except Exception as e:
        log_error(f"Unexpected error in prediction: {str(e)}")
        return jsonify({
            'error': 'Internal server error',
            'reference_id': generate_error_reference()
        }), 500
```



3.8.3 Design for Non-Functional Requirements

Security Design:

- Authentication: JWT-based authentication with refresh token rotation

- Authorization: Role-based access control matrix defining permissions per role

- Data Protection: Field-level encryption for sensitive data, anonymization before storage

- Input Validation: Multi-layer validation (client-side, server-side, database constraints)

- Audit Trail: Immutable audit logs with cryptographic hashing for integrity verification

Performance Design:

- Caching Strategy: Redis cache for frequent queries, model predictions (with appropriate invalidation)

- Database Optimization: Appropriate indexes, query optimization, connection pooling

- Async Processing: Celery workers for background tasks (report generation, analytics updates)

- CDN Usage: Static assets served via CDN for reduced latency

- Load Balancing: Horizontal scaling with load balancer for high availability

Reliability Design:

- Fault Tolerance: Graceful degradation when non-essential services fail

- Redundancy: Database replication, redundant application servers

- Backup Strategy: Automated daily backups with weekly recovery tests

- Circuit Breakers: Fail-fast pattern for external dependencies

Maintainability Design:

- Modular Architecture: Clear separation of concerns, well-defined interfaces

- Configuration Management: Externalized configuration, environment-specific settings

- Testing Strategy: Unit tests, integration tests, end-to-end tests, performance tests

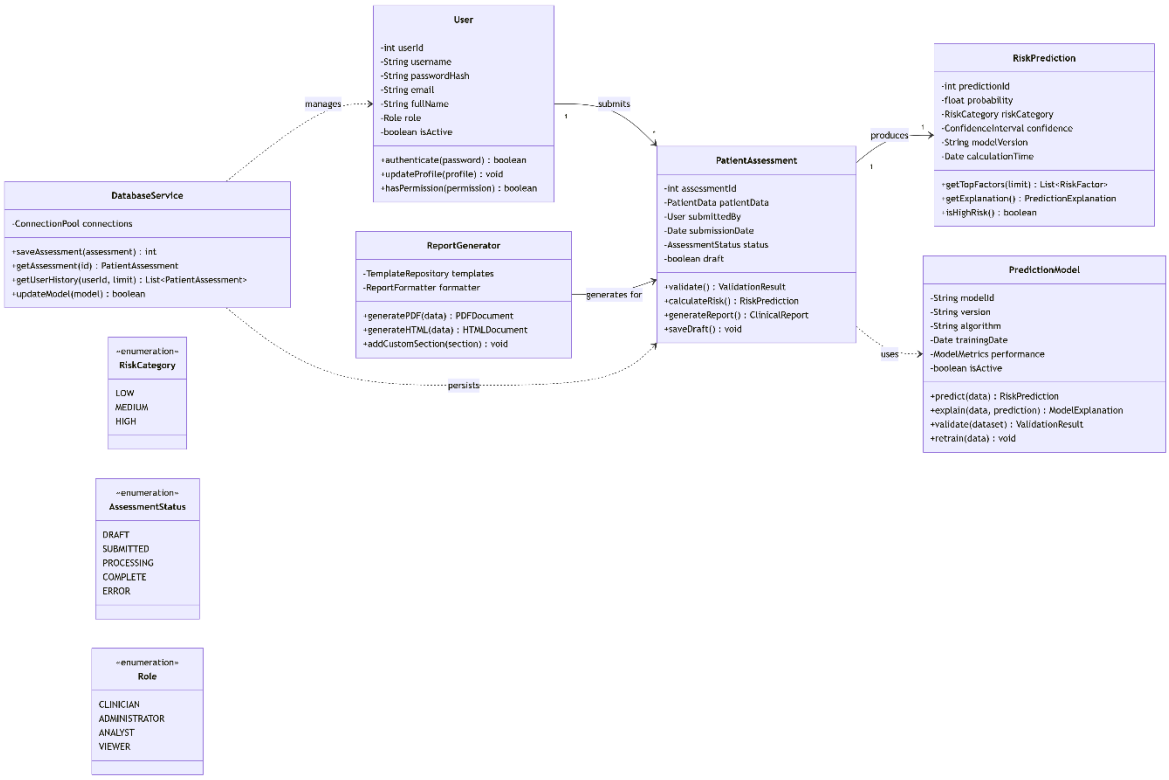
- Documentation: API documentation (OpenAPI/Swagger), architecture decision records

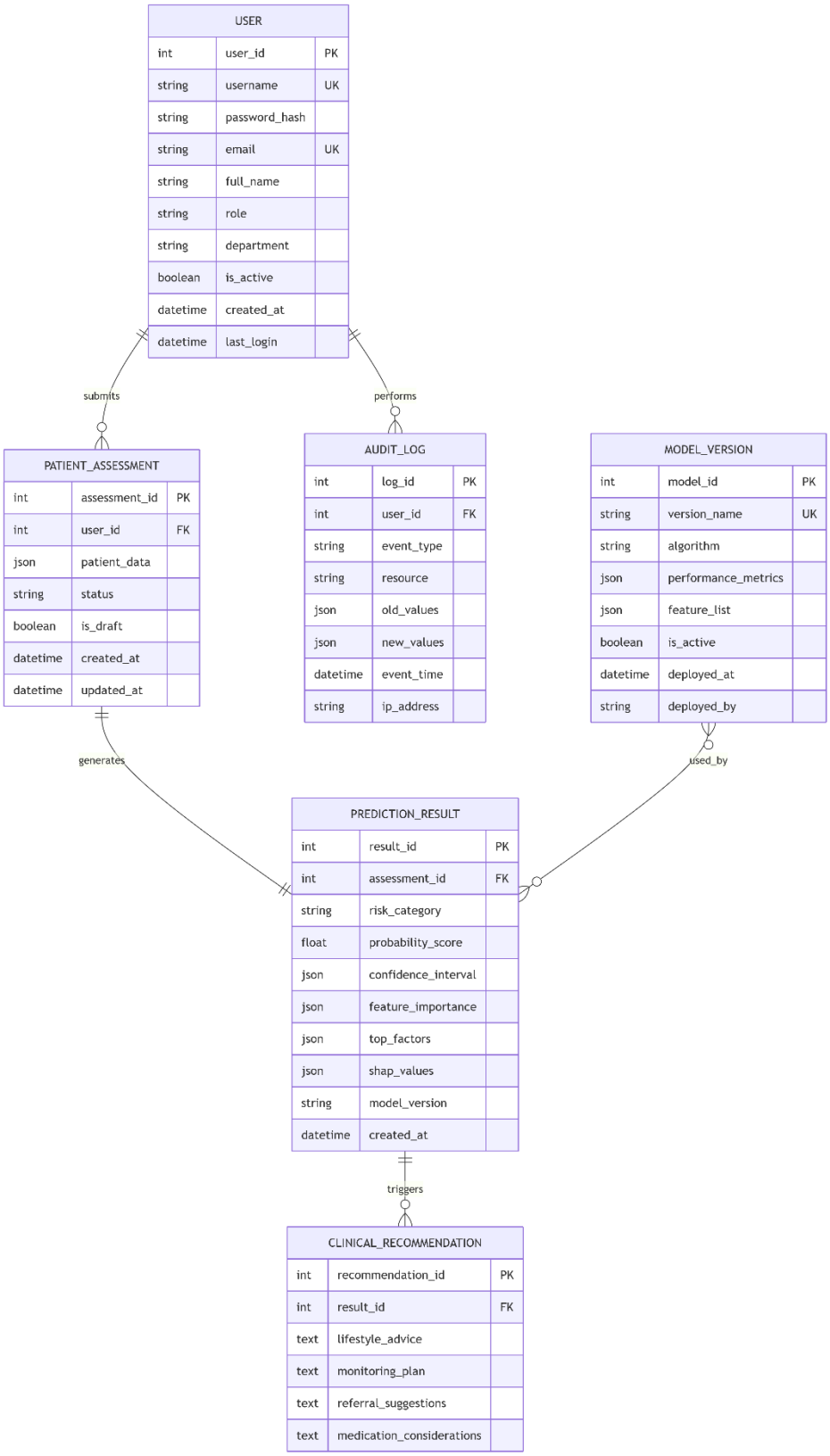
3.9 Physical Design

3.9.1 Database Design

The system uses **SQlite** as the backend relational database management system. The database schema is designed to ensure data integrity, support secure clinical data handling, enable efficient querying, and provide auditability for accountability and compliance.

Enhanced Database Schema:




USER
int user_id PK
string username UK
string password_hash
string email UK
string full_name
string role
string department
boolean is_active
datetime created_at
datetime last_login
submits
performs
PATIENT_ASSESSMENT
int assessment_id PK
int user_id FK
json patient_data
string status
boolean is_draft
datetime created_at
datetime updated_at
AUDIT_LOG
int log_id PK
int user_id FK
string event_type
string resource
json old_values
json new_values
datetime event_time
string ip_address
MODEL_VERSION
int model_id PK
string version_name UK
string algorithm
json performance_metrics
json feature_list
boolean is_active
datetime deployed_at
string deployed_by
generates
used_by
PREDICTION_RESULT
int result_id PK
int assessment_id FK
string risk_category
float probability_score
json confidence_interval
json feature_importance
json top_factors
json shap_values
string model_version
datetime created_at
triggers
CLINICAL_RECOMMENDATION
int recommendation_id PK
int result_id FK
text lifestyle_advice
text monitoring_plan
text referral_suggestions
text medication_considerations

Users Table

The users table stores authentication and authorization information for system users.

Purpose:

- Manage user authentication
- Support role-based access control
- Track account activity and security events

```
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    full_name VARCHAR(100) NOT NULL,
    role VARCHAR(20) NOT NULL CHECK (role IN ('clinician', 'administrator')),
    department VARCHAR(50),
    license_number VARCHAR(50),
    is_active BOOLEAN DEFAULT TRUE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP,
    password_changed_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    failed_login_attempts INTEGER DEFAULT 0,
    account_locked_until TIMESTAMP
);
```

Patient Assessments Table

The patient assessments table captures patient demographic, clinical, and lifestyle data used for coronary artery disease risk assessment.

Purpose:

- Store assessment data entered by clinicians

- Enforce clinical data validation
- Support draft and finalized records

```
CREATE TABLE patient_assessments (
    assessment_id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(user_id) ON DELETE SET NULL,
    assessment_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    age INTEGER NOT NULL CHECK (age BETWEEN 18 AND 120),
    sex VARCHAR(10) NOT NULL CHECK (sex IN ('male', 'female', 'other')),
    total_cholesterol DECIMAL(5,2),
    hdl_cholesterol DECIMAL(5,2),
    ldl_cholesterol DECIMAL(5,2),
    systolic_bp INTEGER,
    diastolic_bp INTEGER,
    fasting_blood_sugar DECIMAL(5,2),
    hba1c DECIMAL(4,2),
    smoking_status VARCHAR(20),
    smoking_years INTEGER,
    cigarettes_per_day INTEGER,
    alcohol_consumption VARCHAR(20),
    physical_activity_level VARCHAR(20),
    diabetes_status BOOLEAN,
    hypertension_status BOOLEAN,
    family_history_cad BOOLEAN,
    previous_cad_event BOOLEAN,
    clinical_notes TEXT,
    draft_status BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT valid_bp CHECK (systolic_bp > diastolic_bp)
```

```
);
```

Prediction Results Table

The prediction results table stores machine learning model outputs and explainability information.

Purpose:

- Persist CAD risk predictions
- Support explainable AI for clinical decision-making

```
CREATE TABLE prediction_results (
    result_id SERIAL PRIMARY KEY,
    assessment_id INTEGER UNIQUE REFERENCES patient_assessments(assessment_id) ON DELETE CASCADE,
    risk_category VARCHAR(10) NOT NULL,
    probability_score DECIMAL(5,4) NOT NULL,
    model_version VARCHAR(50) NOT NULL,
    model_type VARCHAR(50) NOT NULL,
    feature_importance JSONB,
    shap_values JSONB,
    recommendations TEXT[],
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Audit Logs Table

The audit logs table records system events for security and accountability purposes.

Purpose:

- Track user actions
- Support auditing and compliance

```
CREATE TABLE audit_logs (
    log_id SERIAL PRIMARY KEY,
    user_id INTEGER REFERENCES users(user_id),
    event_type VARCHAR(50),
    event_timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ip_address INET,
    action_performed VARCHAR(50),
    status VARCHAR(20)
);
```

Model Versions Table

The model versions table manages deployed and historical machine learning models.

```
CREATE TABLE model_versions (
    model_id SERIAL PRIMARY KEY,
    version_name VARCHAR(50) UNIQUE NOT NULL,
    model_type VARCHAR(50) NOT NULL,
    performance_metrics JSONB,
    model_file_path VARCHAR(255) NOT NULL,
    is_active BOOLEAN DEFAULT FALSE,
    deployed_at TIMESTAMP
);
```

Database Views

A database view is used to support dashboard reporting and analytics.

```
CREATE VIEW vw_dashboard_metrics AS
SELECT DATE(pa.assessment_date) AS assessment_date,
       COUNT(*) AS total_assessments,
       AVG(pr.probability_score) AS average_risk
```

```
FROM patient_assessments pa
JOIN prediction_results pr ON pa.assessment_id = pr.assessment_id
WHERE pa.draft_status = FALSE
GROUP BY DATE(pa.assessment_date);
```

Database Security:

- Role-based privileges: Different database roles for application user, admin user, and read-only analyst

- Row-level security: Policies to ensure users only see their own draft assessments

- Encryption: Transparent data encryption for sensitive columns

- Backup encryption: Encrypted backups with separate key management

3.9.2 User Interface Design

Design Principles:

- Clarity over cleverness: Prioritize clear information presentation over decorative elements

- Consistency: Maintain consistent interaction patterns throughout the interface

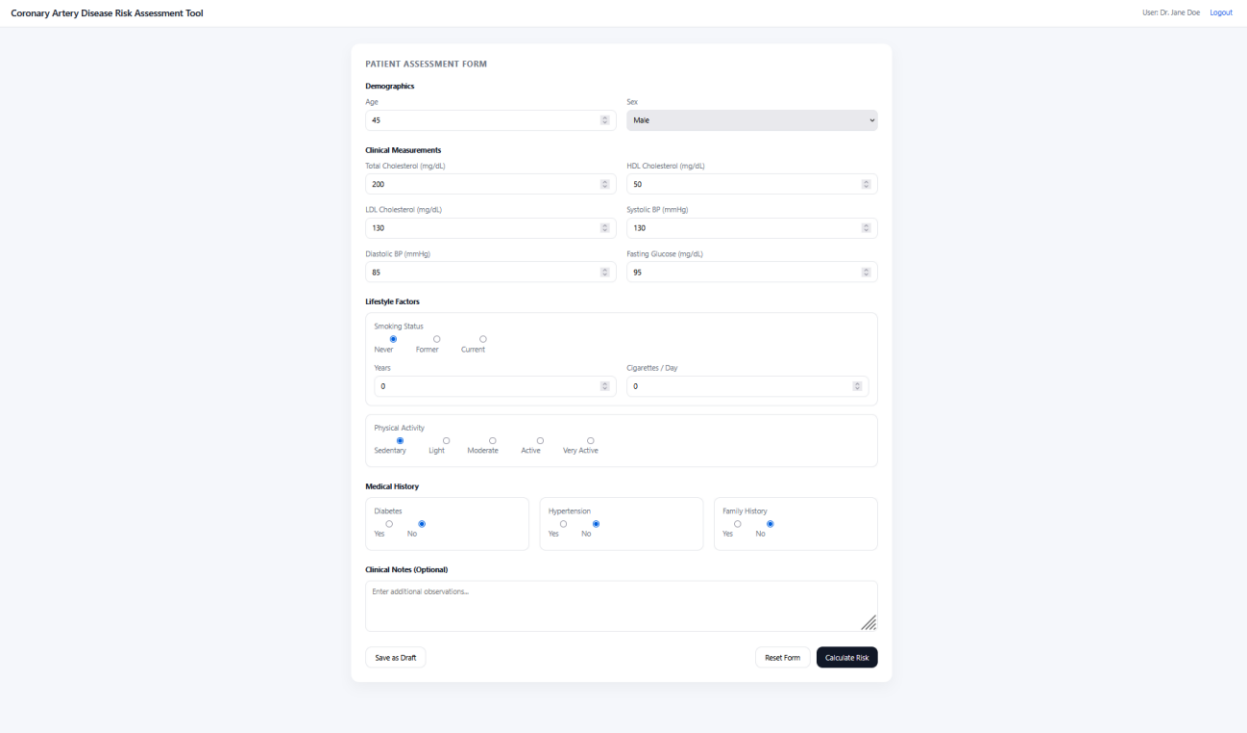
- Progressive disclosure: Show basic information first, details on demand

- Accessibility: Ensure interface is usable by people with diverse abilities

- Responsiveness: Optimize interface for different screen sizes and devices

Wireframe Specifications:

Wireframe A: Enhanced Data Input Form



Key Features of Input Form:

- Progressive Disclosure: Advanced fields (LDL, detailed smoking history) initially hidden, revealed based on basic inputs

- Real-time Validation: Immediate feedback for out-of-range values with suggested normal ranges

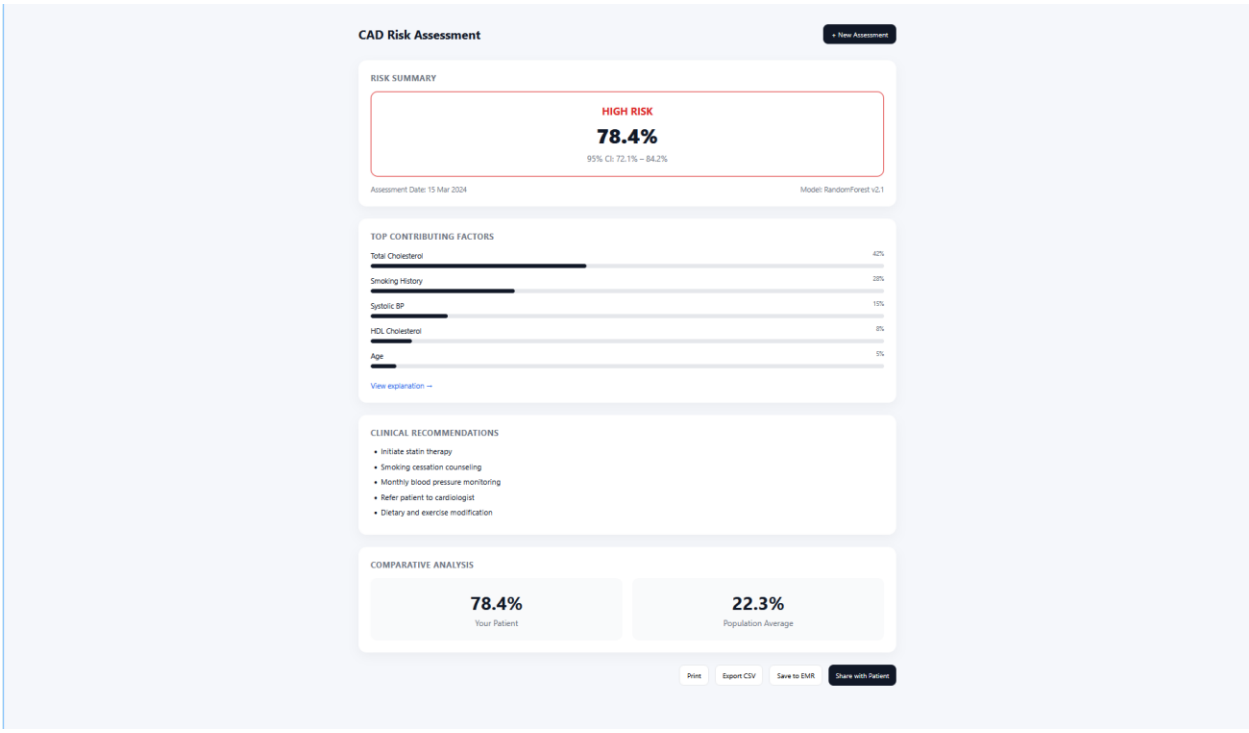
- Contextual Help: Question mark icons providing brief explanations of each parameter

- Unit Display: Clear display of measurement units (mg/dL, mmHg)

- Default Values: Intelligent defaults based on population averages

- Keyboard Navigation: Tab sequence optimized for efficient data entry

Wireframe B: Comprehensive Results Dashboard



Key Features of Results Dashboard:

- Visual Hierarchy: Clear prioritization of risk category with color coding (red for high, yellow for medium, green for low)

- Interactive Elements: Hover-over details for each contributing factor showing exact values and impact

- Comparative Context: Benchmarking against population averages for similar demographics

- Action-Oriented Design: Clear next-step recommendations with reference to clinical guidelines

- Multi-format Output: Options for different output formats based on use case

- Historical Context: Link to view this patient's previous assessments (if available)

Additional Interface Components:

Wireframe C: Historical Assessments View

- Timeline visualization of risk progression over time

- Filtering by date range, risk category, or specific parameters

- Comparison view showing changes in key parameters between assessments

- Export functionality for quality assurance and audit purposes

Wireframe D: Administrator Dashboard

- System health metrics (uptime, response times, error rates)

- User activity reports

- Model performance monitoring (accuracy drift over time)

- Data quality dashboard showing completeness and distribution of entered parameters

Accessibility Considerations:

- Screen Reader Support: Proper ARIA labels and semantic HTML structure

- Keyboard Navigation: All functionality accessible via keyboard

- Color Contrast: Minimum contrast ratio of 4.5:1 for normal text

- Text Resizing: Support for browser text zoom up to 200%

- Alternative Text: Descriptive alt text for all informative images

Responsive Design Breakpoints:

- Mobile (<768px): Stacked form layout, simplified visualizations, touch-friendly controls

- Tablet (768px-1024px): Two-column form layout, moderate detail visualizations

- Desktop (>1024px): Multi-column form layout, detailed visualizations with side-by-side comparisons

3.9.3 Deployment Architecture

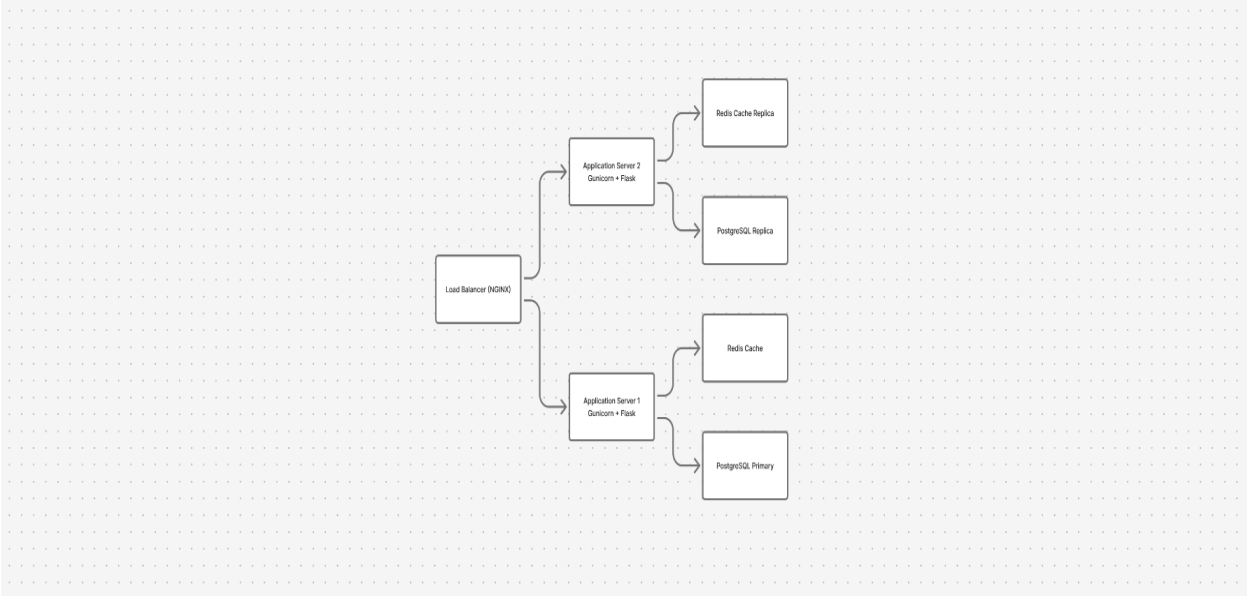
Development Environment:

- Local Development: Docker Compose setup with all services (PostgreSQL, Redis, Flask app)

- CI/CD Pipeline: GitHub Actions for automated testing and deployment

- Testing Environment: Staging environment mirroring production configuration

Production Environment:



Monitoring Stack:

- Application Monitoring: Prometheus metrics with Grafana dashboards

- Log Management: Centralized logging with ELK stack (Elasticsearch, Logstash, Kibana)

- Uptime Monitoring: External health checks from multiple geographic locations

- Business Metrics: Custom dashboards tracking assessment volume, risk distribution, user engagement

Disaster Recovery:

- Backup Strategy: Daily full backups with hourly transaction log backups
- Recovery Point Objective (RPO): 1 hour maximum data loss
- Recovery Time Objective (RTO): 4 hours for full system restoration
- Geographic Redundancy: Option to deploy to secondary region for critical applications