Loan Automation Business Analysis Project

1. Project Introduction & Background

ABCD Bank, a fictional but realistically modeled mid-sized Canadian financial institution, has traditionally relied on manual workflows to process personal loan applications. These workflows involve human underwriters, document reviews, manual credit checks, and policy-based approvals that vary based on employee judgment and workload. The bank now faces pressure from competitive lenders who offer near-instant lending decisions through automated systems. Customers increasingly expect fast, transparent digital loan experiences without long waiting periods or complex document requests.

To remain competitive and reduce operational friction, ABCD Bank seeks to implement an **Automated Loan Decisioning System (ALDS)**. This new system will automate income verification, credit score retrieval, risk scoring, and initial decisioning. Human underwriters will still be involved in edge-case or complex scenarios, but the majority of applications will be completed digitally, significantly reducing workload and turnaround time.

The goal of this project is to design a full business analysis package, including requirements, workflows, user stories, business rules, and a complete BRD and FSD set. All data, processes, personas, and assumptions have been constructed to reflect authentic business analysis practices within the banking domain.

Problem: Analytics show that **30% of applicants abandon their application before approval**, which translates to millions in lost revenue. Loan approval takes 7 days.

Desired Value: Improve the approval speed and reduce customer drop-off by automating income verification, risk scoring, and document validation.

---

2. Project Objectives

1. Reduce personal loan approval time from the current 5–7 days to under 1 hour for standard applications.
2. Improve consistency in underwriting by using a centralized rules engine and standardized decision logic.
3. Minimize manual document collection through external API integrations for income and bank account verification.
4. Strengthen compliance alignment with OSFI B-20 and anti-money laundering (AML) requirements.
5. Increase customer satisfaction through a clearer, more intuitive digital application experience.
6. Reduce the operational burden on underwriters by automating predictable and low-risk cases.

7. Establish a modern technical architecture that can be extended to future lending products.
8. Enhance auditability and risk transparency through automated logs, decision trails, and standardized workflows.
9. Provide clear value to the business through reduced drop-off rates and higher conversion.
10. Prepare ABCD Bank for long-term digital modernization in lending.

---

3. Stakeholder Personas

3.1 Alireza – Senior Underwriter

Olivia has over 12 years of underwriting experience. She is deeply familiar with credit bureau reports, debt-to-income calculations, and mitigating factors for borderline applicants. She spends much of her time evaluating income documents and manually calculating ratios and risk scores. Her biggest frustration is unclear or incomplete applicant information. She wants an automated system that removes repetitive work while still allowing her to intervene in complex cases where human judgment is essential.

3.2 David – Compliance Officer

David's chief concern is adherence to regulatory requirements. He ensures the loan process follows OSFI guidelines, KYC/AML laws, and proper customer identity verification. He needs clear audit trails, timestamps, decision logs, and consistent application of rules. David dislikes unpredictable manual decisions that differ between underwriters. He supports automation because it brings uniformity and traceability.

3.3 Priya – Solutions Architect

Priya oversees the bank's technology landscape. She prefers loosely coupled systems that communicate through well-defined APIs. She wants the loan automation system to integrate with existing systems, including core banking and document management. She focuses on stability, uptime, error handling, and predictable system behavior. Her concerns revolve around scalability, vendor dependency, and integration constraints.

3.4 Gomez – Lending Product Manager

Marco focuses on customer experience and product profitability. He's concerned about drop-off during the loan application, customer confusion, and long approval times leading applicants to competitors. He wants a smooth, modern application flow and a system that increases approved applications without raising risk exposure.

3.5 Julia – Call Centre Team Lead

Julia manages the front-line support team. When customers are confused or stuck, they call her centre. She wants a decisioning system that provides clear, understandable status updates, so her team doesn't spend time trying to decode the back-office processes.

4. Assumptions

1. All integrations with income verification providers use REST APIs with encrypted data transmission.
2. Credit bureau data is accessible in real time through a secure connection.
3. Applicants have a stable internet connection during their application.
4. The bank's internal staff can adapt to modest training on the new system.
5. Underwriters remain responsible for special-case and escalated applications.
6. Business rules align with OSFI regulations and standard risk management practices.
7. All data stored within the system complies with Canadian privacy laws.
8. The project team has full access to mock data and testing environments.
9. Applicants submit complete applications digitally.
10. No major regulatory changes occur during the project timeline.

5. Business Requirements

5.1Functional Requirements

1. FR-001

Requirement: The system SHALL retrieve applicant credit scores and credit reports from designated credit bureau APIs (Equifax and TransUnion).

Priority: MUST

Rationale: Credit bureau data is required to assess applicant risk and make consistent lending decisions.

Dependencies: Credit bureau contracts, API credentials, consent capture in UI.

Verification method: End-to-end credit pull integration test and monitoring dashboard showing successful pulls per provider.

Owner: Integrations/Platform Team

1.1. FR-001.1

Requirement: The system SHALL support both Equifax and TransUnion endpoints.

Priority: MUST

Rationale: Redundancy and broader coverage improves data availability and accuracy.

Dependencies: API endpoints from both bureaus; mapping logic.

Verification method: Provider-specific pull tests for both vendors.

Owner: Integrations Team

1.2 FR-001.2

Requirement: On credit-bureau API failure, the system SHALL retry twice using exponential backoff and then follow the defined escalation flow.

Priority: MUST

Rationale: Resilience reduces false negatives and manual rework.

Dependencies: Retry library/config, alerting system, failed-pull queue.

Verification method: Fault injection test simulating provider outage and verifying retries and escalation logs.

Owner: Platform/DevOps

2. FR-002

Requirement: The system SHALL calculate debt-to-income (DTI) ratios using applicant inputs and validated income sources.

Priority: MUST

Rationale: DTI is a core risk metric for underwriting decisions.

Dependencies: Income verification provider, applicant debt inputs, calculation engine.

Verification method: Unit and integration tests for DTI formula across test cases; calculation audit logs.

Owner: Credit Risk / Product

2.1. FR-002.1

Requirement: The system SHALL include all revolving and installment debts in the DTI calculation.

Priority: MUST

Rationale: Comprehensive debt inclusion ensures accurate risk scoring.

Dependencies: Debt data sources, mapping logic.

Verification method: Data integrity tests and reconciliation reports.

Owner: Data Engineering

2.2. FR-002.2

Requirement: The system SHALL flag applications when DTI exceeds configured threshold values.

Priority: MUST

Rationale: Flags enable correct routing and risk treatment.

Dependencies: Rules engine and configurable threshold store.

Verification method: Rules engine test runs and flagging metrics in monitoring.

Owner: Risk / Underwriting

3. FR-003

Requirement: The system SHALL assign risk categories (Low, Medium, High) via the centralized rules engine using credit, DTI, and other inputs.

Priority: MUST

Rationale: Standardized risk categories ensure consistent decisioning and reporting.

Dependencies: Rules engine, input data feeds, rule repository.

Verification method: Rules engine audit logs showing rule firings and test-case coverage for each category.

Owner: Risk

3.1. FR-003.1

Requirement: The system SHALL maintain an auditable trail of rules fired and inputs used for each risk assessment.

Priority: MUST

Rationale: Traceability for compliance and audit.

Dependencies: Audit log storage, immutable logging mechanism.

Verification method: Log retrieval and integrity checks; sample audit review.

Owner: Compliance / Risk

4. FR-004

Requirement: The system SHALL auto-approve applications categorized as Low risk if minimum credit score and DTI thresholds are satisfied.

Priority: MUST

Rationale: Automating low-risk approvals reduces turnaround time and operational cost.

Dependencies: Rules engine, approval artifacts generation, audit logging.

Verification method: Workflow tests for auto-approval path and audit log entries showing decision rationale.

Owner: Product / Risk

4.1. FR-004.1

Requirement: The system SHALL enforce constraints that only eligible applicants per the thresholds may be auto-approved.

Priority: MUST

Rationale: Prevents unauthorized approvals and risk leakage.

Dependencies: Threshold config management and enforcement logic.

Verification method: Threshold change test and negative test cases.

Owner: Risk

5. FR-005

Requirement: The system SHALL route Medium-risk applications to the underwriting work queue for manual review.

Priority: MUST

Rationale: Human judgment is required for borderline cases to balance risk and customer service.

Dependencies: Underwriting queue system, notification to underwriters.

Verification method: End-to-end scenario tests verifying queue population and notification.

Owner: Underwriting / Operations

5.1. FR-005.1

Requirement: If fraud indicators are detected, the system SHALL escalate the case to Fraud Operations rather than the standard underwriting queue.

Priority: MUST

Rationale: Speed and containment of potential fraud incidents.

Dependencies: Fraud detection subsystem, escalation flow.

Verification method: Fraud detection alert tests and escalation audit trails.

Owner: Fraud Ops / Risk

6. FR-006

Requirement: The system SHALL auto-decline applications that do not meet the minimum eligibility thresholds.

Priority: MUST

Rationale: Protects the institution by ensuring consistent enforcement of baseline criteria.

Dependencies: Rules engine and decline template generator.

Verification method: Negative path tests and generated decline artifacts validation.

Owner: Risk

6.1. FR-006.1

Requirement: The system SHALL automatically generate standardized decline letters for auto-declines.

Priority: MUST

Rationale: Provides customers with compliant communication and improves transparency.

Dependencies: Template management and document generation service.

Verification method: Template rendering tests and version logs.

Owner: Operations / Legal

7. FR-007

Requirement: The system SHALL store all decision logs in an immutable, timestamped format for auditability.

Priority: MUST

Rationale: Regulatory and compliance requirements mandate lasting traceability of decisions.

Dependencies: Immutable log store (WORM or equivalent), retention policy.

Verification method: Log integrity checks and retention verification.

Owner: Compliance / Security

8. FR-008

Requirement: The system SHALL validate government-issued identification via an approved third-party identity verification (KYC) provider.

Priority: MUST

Rationale: KYC compliance is mandatory for onboarding and risk mitigation.

Dependencies: KYC provider contracts, consent capture, secure transmission.

Verification method: Integration tests with provider and verification status reporting.

Owner: Compliance / Integrations

9. FR-009

Requirement: The system SHALL provide applicants with real-time status updates via push notifications and an in-app dashboard.

Priority: SHOULD

Rationale: Improves customer experience and reduces call center load.

Dependencies: Notification service, eventing pipeline, UI dashboard.

Verification method: Event-driven test scripts and notification delivery metrics.

Owner: Product / Customer Experience

10. FR-010

Requirement: The system SHALL generate standardized approval and decline letters using versioned templates that include regulatory disclosures.

Priority: MUST

Rationale: Ensures consistent, compliant customer communication.

Dependencies: Template repository, version control, document generation engine.

Verification method: Template rendering tests and template version audit logs.

Owner: Legal / Operations

5.2 Non-Functional Requirements

1. NFR-001

Requirement: The system SHALL process standard loan applications end-to-end in under 2 minutes under normal operating conditions.
Priority: MUST
Rationale: Performance target supports the business goal of sub-hour approvals.
Dependencies: API SLAs, processing engine, network throughput.
Verification method: Performance testing under representative load; 95th-percentile timing report.
Owner: Platform / SRE

2. NFR-002
Requirement: The system SHALL maintain 99.5% monthly uptime.
Priority: MUST
Rationale: Availability is critical for customer trust and regulatory expectations.
Dependencies: Redundant architecture, monitoring and failover mechanisms.
Verification method: Availability monitoring reports and incident postmortems.
Owner: SRE / Infrastructure

3. NFR-003
Requirement: The system SHALL support at least 500 concurrent application sessions without functional degradation.
Priority: SHOULD
Rationale: Ensures acceptable performance during peak application volumes.
Dependencies: Autoscaling policies, load balancers.
Verification method: Load testing with concurrency targets and SLA reporting.
Owner: Platform

4. NFR-004
Requirement: Data at rest SHALL be encrypted with AES-256 or equivalent.
Priority: MUST
Rationale: Protects sensitive customer data and meets regulatory requirements.
Dependencies: Storage encryption capabilities, key management service.
Verification method: Security configuration audit and encryption verification.
Owner: Security

5. NFR-005
Requirement: Data in transit SHALL use TLS 1.2 or higher.
Priority: MUST

Rationale: Network encryption protects data during transmissions.
Dependencies: TLS configuration and certificate management.
Verification method: Network security scans and cipher-suite verification.
Owner: Security / Network

6. NFR-006
Requirement: The system SHOULD support multilingual applicants (English and French minimum).
Priority: SHOULD
Rationale: Required for Canadian market inclusivity and regulatory expectations in some jurisdictions.
Dependencies: Localization framework and translation assets.
Verification method: Localization QA and UI checks.
Owner: Product / UX

7. NFR-007
Requirement: The system SHALL retain detailed logs for a minimum of seven years in accordance with retention policy.
Priority: MUST
Rationale: Meets regulatory and audit retention requirements.
Dependencies: Long-term storage, retention policy enforcement.
Verification method: Retention policy audit and spot checks.
Owner: Compliance / IT

8. NFR-008
Requirement: Integrations SHALL withstand peak traffic without degradation and include retry/backoff and alerting.
Priority: MUST
Rationale: Ensures operational resilience and consistent customer experience.
Dependencies: Integration patterns, SLAs with vendors.
Verification method: Integration load tests and alerting test scenarios.
Owner: Integrations / SRE

9. NFR-009
Requirement: The user interface SHOULD load pages within 2 seconds for typical user connections.
Priority: SHOULD
Rationale: UX performance directly impacts drop-off and conversion.
Dependencies: CDN, frontend optimization, backend response times.
Verification method: Frontend performance metrics and synthetic monitoring.
Owner: Frontend / UX

10. NFR-010
Requirement: The system SHALL comply with WCAG AA accessibility guidelines.
Priority: MUST
Rationale: Accessibility compliance and inclusive customer access.
Dependencies: Accessible UI components and QA processes.
Verification method: Accessibility audit and remediation report.
Owner: UX / Accessibility Lead

11. NFR-011
Requirement: The system SHALL support versioning for decision rules and document templates for auditability.
Priority: MUST
Rationale: Enables traceability and rollback for regulatory audits.
Dependencies: Rules repository, template version control.
Verification method: Version history audit and rollback test.
Owner: Product / Risk

12. NFR-012
Requirement: The system SHALL maintain SLA compliance with external API providers per contractual agreements.
Priority: MUST
Rationale: SLA breaches can impact performance and legal exposure.
Dependencies: Vendor contracts and monitoring.
Verification method: SLA dashboards and breach reports.
Owner: Vendor Management / Integrations

13. NFR-013
Requirement: The system SHALL include monitoring and alerting for failed transactions and degraded performance.
Priority: MUST
Rationale: Early detection reduces business impact and supports SRE processes.
Dependencies: Observability stack and alerting rules.
Verification method: Alert firing tests and incident response exercises.
Owner: SRE / Platform

14. NFR-014
Requirement: The system SHALL support rollback and error recovery mechanisms to maintain data integrity following partial failures.
Priority: MUST
Rationale: Recovery capabilities reduce operational risk and prevent data corruption.
Dependencies: Transactional patterns, compensation workflows, backups.
Verification method: Disaster recovery and rollback testing.

Owner: Platform / DBAs

NFR-015
Requirement: The system SHALL maintain data consistency across multi-system integrations (defined consistency model to be specified — eventual or strong where required).
Priority: MUST
Rationale: Accurate cross-system data is essential for correct decisioning and reporting.
Dependencies: Integration design patterns, idempotency, reconciliation jobs.
Verification method: End-to-end reconciliation tests and consistency checks.
Owner: Platform / Data Engineering

---

6. Business Rules

1. Applicants must be Canadian residents with valid identification.
2. Minimum credit score for automatic approval is 680.
3. Applicants with credit scores under 580 shall be automatically declined.
4. DTI ratio must not exceed 43% for auto-approval.
5. For DTI between 44% and 55%, cases must route to manual review.
6. Employment must be verified through a valid income data source.
7. Applications flagged for potential fraud must trigger an immediate hold.
8. Applicants with recent bankruptcies (less than 6 years) must be auto-declined.
9. Applicants with stable income and low debt may receive preferential rates.
10. All risk scores and decision factors must be logged with timestamps.
11. Any deviations from automated decisions must be approved and logged.
12. Late document submissions after approval cutoff must trigger review.
13. Data entered by applicants must be validated for format, completeness, and plausibility.
14. Notifications to customers must follow standardized templates.
15. System must respect regulatory hold periods before disbursement.

---

7. As-Is Workflow

1. Applicant fills out a paper or digital form.
2. Applicant uploads or emails income documents.
3. Underwriter manually calculates DTI.
4. Underwriter retrieves credit report.
5. Underwriter evaluates employment stability.
6. Underwriter makes an approval or decline decision.
7. Applicant receives a communication after several days.
8. Call center frequently answers status inquiries.
9. Compliance team performs spot checks on underwriting quality.
10. Final documentation is prepared manually and emailed.

8. To-Be Workflow

1. Applicant completes online form.
2. Income API retrieves validated employment and income data.
3. Credit bureau API provides instant credit information.
4. Rules engine evaluates all metrics and assigns a risk score.
5. Auto-approval or auto-decline is triggered based on thresholds.
6. Medium-risk cases route to underwriters.
7. Applicant receives near-instant status updates.
8. System creates audit logs.
9. Approved applicants receive a digital agreement.
10. Loan is disbursed automatically upon acceptance.

This future state reduces manual work by 60–70% and dramatically shortens turnaround time.

---

9. User Stories and Acceptance Criteria

1. Applicant – Automatic Income Verification
   As an applicant, I want my income to be verified automatically so I don't need to upload documents.

   Acceptance Criteria:
   - The system retrieves verified income data from the income API within 5 seconds.
   - Retrieved income values auto-populate the applicant's form fields.
   - Any mismatch between declared income and verified income triggers a warning flag.
   - If the income API is unavailable, the applicant sees a clear retry message.
   - Income verification requests are logged with timestamp and data source.

2. Underwriter – Medium-Risk Routing.
   As an underwriter, I want medium-risk cases routed to me so I can focus on judgment-based decisions.

   Acceptance Criteria:
   - Applications classified as medium-risk appear automatically in the underwriting work queue.
   - The system routes low- and high-risk applications to their respective automated paths.

- The underwriter can view detailed risk rationale (e.g., DTI, credit score, income verification).
- Manual escalation remains possible for unclear or edge-case scenarios.
- Routing actions are fully logged for audit and compliance.

3. Compliance Officer – Decision Logs.
   As a compliance officer, I want decision logs stored securely so audits are consistent and complete.

   Acceptance Criteria:
   - Every automated and manual decision generates an immutable log entry.
   - Logs capture timestamp, data used, rules fired, and decision outcome.
   - Logs are stored securely following the bank's retention and encryption standards.
   - Compliance users can retrieve logs through a secure interface.
   - Any modification attempt triggers alerts or is blocked entirely.

4. Architect – Standardized APIs.
   As an architect, I want APIs to follow standard patterns so integrations are predictable.

   Acceptance Criteria:
   - All APIs follow REST conventions, including standardized authentication and error codes.
   - API contracts remain version-controlled with backward compatibility.
   - API failures return predictable, structured error responses.
   - System integrations support retries with exponential backoff.
   - API response times meet defined performance SLAs

5. Product Manager – Real-Time Reporting.
   As a product manager, I want real-time reporting so I can track drop-off and conversion.

   Acceptance Criteria:
   - Metrics for each stage of the loan application flow update in real time.
   - Reports show conversion, drop-off, approval rates, decline rates, and API failures.
   - Filters allow segmentation by product, channel, and risk tier.
   - Reporting dashboards load within 2 seconds.
   - Data accuracy stays within defined tolerance levels.

6. Applicant – Instant Status Updates.
   As an applicant, I want instant status updates so I'm not confused about progress.

   Acceptance Criteria:
   - Status changes trigger immediate notifications in the portal.
   - The system reflects real-time states such as "In Review," "Approved," "Declined," and "More Info Needed."
   - Applicants never see internal underwriting notes or raw bureau data.
   - Error states provide clear instructions instead of generic messages.
   - Status updates align with logged system events.

7. Underwriter – Auto-Calculated Risk Metrics.
   As an underwriter, I want auto-calculated risk metrics so I don't waste time on math.

   Acceptance Criteria:
   - The system automatically calculates DTI, risk tier, and credit utilization.
   - Calculations follow standardized business rules.
   - Underwriters can view formulas or data inputs used in the calculation.
   - Any calculation anomaly triggers a system flag.
   - All calculated metrics are stored in the audit log.

8. Call Centre Lead – Clear Status Categories.
   As a call centre lead, I want clear status categories so my team can support callers efficiently.

   Acceptance Criteria:
   - The system provides a simplified, customer-safe status view for call centre agents.
   - Agents can see the reason for holds (e.g., pending verification, manual review).
   - Status categories are mapped to internal workflow states.
   - Agents can escalate unclear cases to underwriting or tech support.
   - All customer inquiries are linked to the application ID in the system.

9.1 Gherkin Version:

```
Feature: Automatic Income Verification
 As an applicant
 I want my income to be verified automatically
 So that I don't need to upload documents

 Scenario: Successful income verification
  Given the applicant submits a loan application
```

```
    When the system retrieves income data from the API within 5 seconds
    Then the applicant’s income fields auto-populate
    And any mismatch between declared and verified income triggers a warning flag
    And the verification request is logged with timestamp and data source

  Scenario: Income API unavailable
    Given the applicant submits a loan application
    When the income API is unavailable
    Then the applicant sees a clear retry message

Feature: Medium-Risk Routing
  As an underwriter
  I want medium-risk cases routed to me
  So that I can focus on judgment-based decisions

  Scenario: Medium-risk application routing
    Given a loan application is evaluated as medium-risk
    When the system processes the application
    Then it appears in the underwriter’s work queue
    And the underwriter can view detailed risk rationale
    And manual escalation is possible
    And routing actions are logged for audit and compliance

Feature: Decision Logs
  As a compliance officer
  I want decision logs stored securely
  So that audits are consistent and complete

  Scenario: Log creation
    Given a loan application is processed
    When a decision (manual or automated) is made
    Then an immutable log entry is created capturing timestamp, data, rules fired, and
outcome
    And logs are stored securely according to retention and encryption standards
    And compliance users can retrieve logs via a secure interface
    And modification attempts are blocked or trigger alerts

Feature: Standardized APIs
  As an architect
  I want APIs to follow standard patterns
  So that integrations are predictable
```

Scenario: API implementation
Given the loan system integrates with external services
When APIs are implemented
Then all APIs follow REST conventions with standardized authentication and error codes
And API contracts are version-controlled with backward compatibility
And failures return structured error responses
And integrations support retries with exponential backoff
And API response times meet defined SLAs

Feature: Real-Time Reporting
As a product manager
I want real-time reporting
So that I can track drop-off and conversion

Scenario: Reporting dashboard
Given loan applications are processed
When metrics are updated
Then the dashboard reflects conversion, drop-off, approval and decline rates, and API failures in real time
And filters allow segmentation by product, channel, and risk tier
And dashboard loads within 2 seconds
And data accuracy is within defined tolerance levels

Feature: Instant Status Updates
As an applicant
I want instant status updates
So that I'm not confused about progress

Scenario: Status notifications
Given a loan application is in process
When the status changes
Then the applicant receives immediate portal notifications
And the system reflects states like "In Review," "Approved," "Declined," or "More Info Needed"
And applicants never see internal notes or raw bureau data
And errors provide clear instructions
And status updates align with logged system events

Feature: Auto-Calculated Risk Metrics
As an underwriter
I want auto-calculated risk metrics
So that I don't waste time on math

```
  Scenario: Risk metrics calculation
    Given applicant data is available
    When the system calculates DTI, risk tier, and credit utilization
    Then calculations follow standardized business rules
    And underwriters can view formulas or inputs used
    And calculation anomalies trigger a system flag
    And all calculated metrics are stored in the audit log

Feature: Clear Status Categories
  As a call centre lead
  I want clear status categories
  So that my team can support callers efficiently

  Scenario: Customer status view
    Given an application is in the system
    When call centre agents view it
    Then they see simplified, customer-safe status categories
    And reasons for holds (pending verification, manual review)
    And categories are mapped to internal workflow states
    And agents can escalate unclear cases
    And all inquiries are linked to the application ID
```

---

11. Functional Specification

11.1 System Components

• Customer Application Portal • Income Verification Service • Credit Bureau Integration Layer • Rules Engine • Underwriting Dashboard • Audit Log Database • Document Generation Service

11.2 Core Functions

The system validates applicant identity, retrieves financial data from external sources, calculates risk metrics using a standardized rules engine, determines automated decisions, and routes special cases to human underwriters. Approved applicants receive agreements, while declined applicants receive standardized notifications.

11.3 Data Fields

• Personal Information: name, DOB, address, phone • Identity Verification: ID number, verification status • Income Details: salary, employment duration, pay frequency • Credit Data: credit score, delinquency history, credit inquiries • Calculated Metrics: DTI ratio, risk category • Decision Data: approved, declined, manual review

11.4 Integrations

The system communicates with external providers over HTTPS using authenticated REST calls. Responses include JSON payloads with income summaries, credit histories, and employment verification details. Failure scenarios rely on retry logic and fallback messaging.

---

12. UAT Test Plan

12.1 Test Objectives

Validate that the automated loan decisioning workflow functions as expected, adheres to business rules, and meets compliance standards.

12.2 Test Scope

Includes application submission, data retrieval, rules-based decisioning, error handling, manual routing, audit logging, and document generation.

12.3 Test Scenarios

• Successful auto-approval • Auto-decline due to low credit score • Manual review routing for borderline cases • API failure for income verification • Incorrect applicant data • Fraud flag scenario • Document generation for approved loans • Audit log validation • Real-time status updates

12.4 Test Data

A full set of mock applicant profiles covering all risk tiers, error cases, and edge conditions.

---

13. Impact Analysis

The new automated system is expected to: • Reduce loan processing time by up to 95%. • Lower underwriting workload by at least 60%. • Increase applicant conversion rate by 20–25%. • Reduce compliance violations through consistent rule enforcement. • Improve customer satisfaction due to faster decisions. • Strengthen competitive positioning in digital lending. • Reduce operational costs associated with manual review.

---

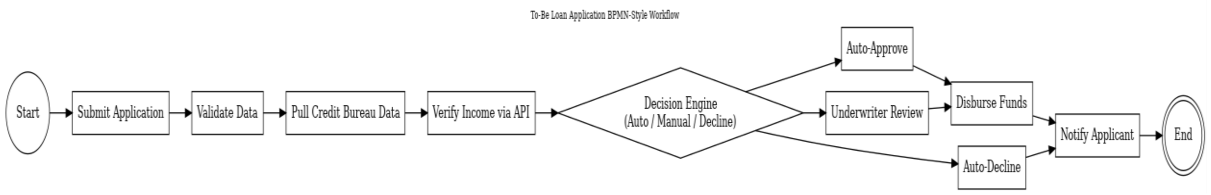
14. Project Summary

This project provides a complete end-to-end business analysis deliverable for an automated loan decisioning system. While based on a fictional bank, it represents real-world banking processes and the full scope of work expected from a skilled business analyst. The outputs include business requirements, workflows, rules, specifications, user stories, and a full UAT test plan. It serves as a comprehensive portfolio project that demonstrates structured analysis, domain knowledge, and solution design for digital lending modernization.

Supplementary Visuals and Diagram Tooling (Industry-Standard)

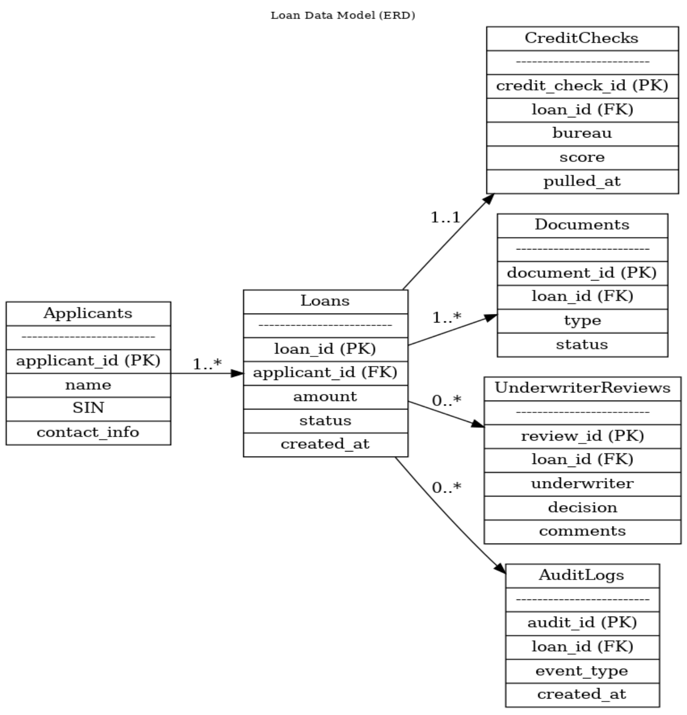
Detailed Workflow Diagram (BPMN) (Visio - Flowable)

A BPMN 2.0 workflow showing the end-to-end customer journey: 1. Customer submits loan application. 2. System validates data. 3. Credit Bureau check. 4. Decision Engine runs automated rules. 5. Exception path: manual underwriter review. 6. Final approval or rejection. 7. Loan disbursement. 8. Notification to user.



ERD (Entity Relationship Diagram) (Lucidchart)

A database-level diagram showing tables such as: - Applicants - Loans - CreditChecks - Documents - UnderwriterReviews - AuditLogs

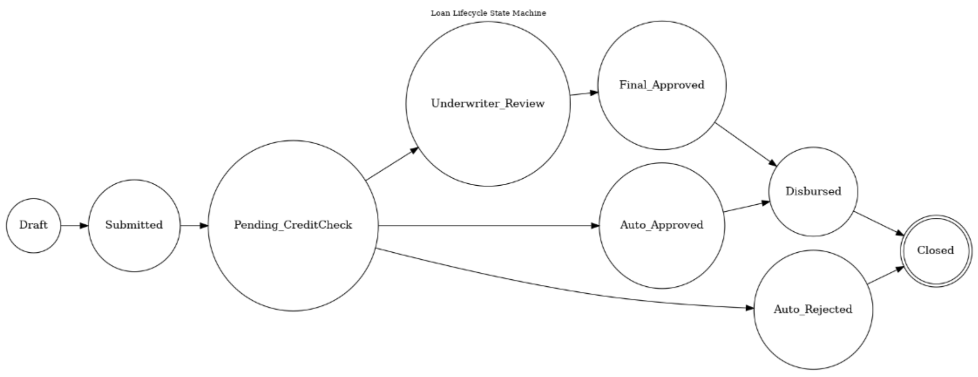


With relationships: - One Applicant → Many Loans - One Loan → One CreditCheck - One Loan → Many Documents - One Loan → Many AuditLogs

---

State Machine Diagram for Loan Lifecycle (Lucidchart)

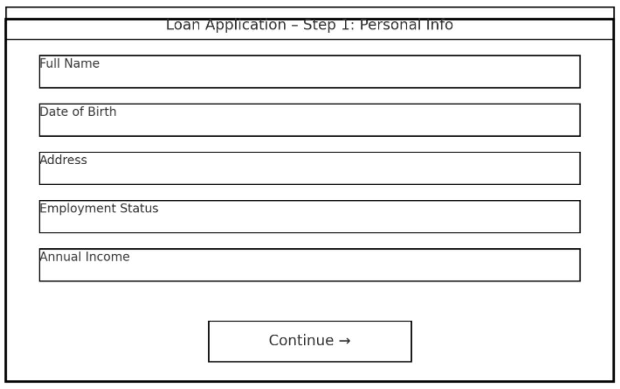
A state diagram showing transitions: - Draft → Submitted → Pending_CreditCheck → Auto_Approved → Auto_Rejected → Underwriter_Review → Final_Approved → Disbursed → Closed.

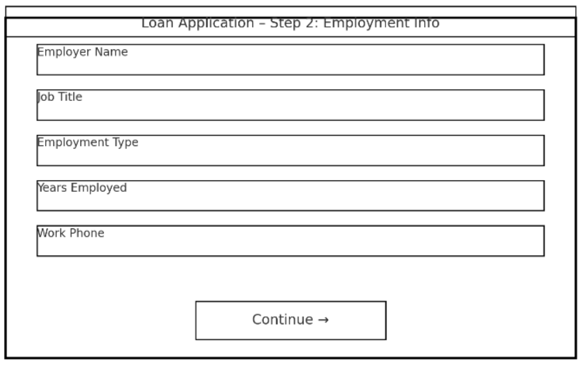


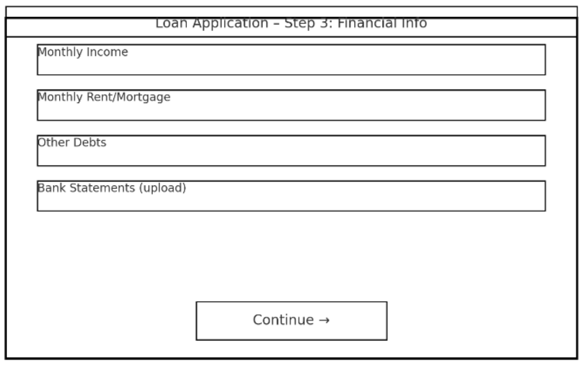
Industry tools I can use instead of Draw.io: **Lucidchart**

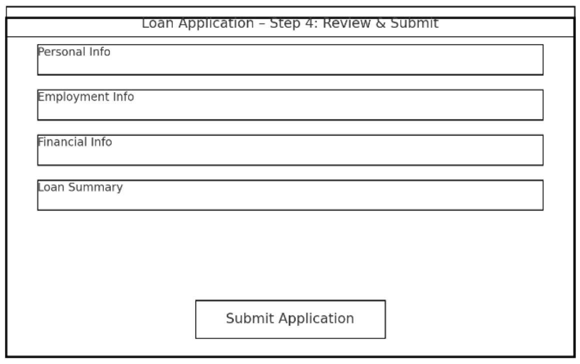
UI Mockups & Wireframes (Figma)

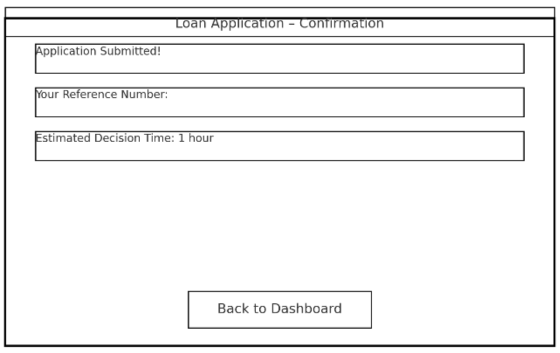
A realistic loan application UI mockup consisting of: - Step 1: Personal Info - Step 2: Employment Info - Step 3: Financial Info - Step 4: Document Upload - Step 5: Review & Submit







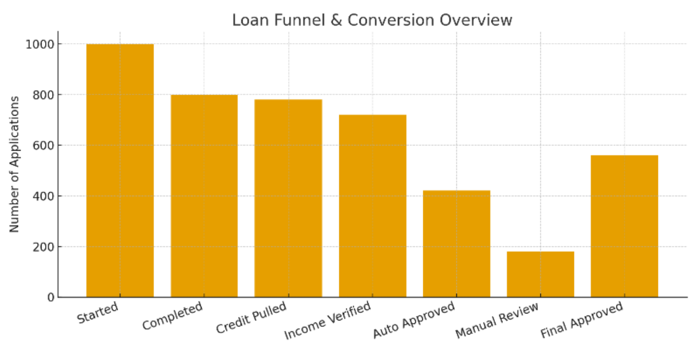


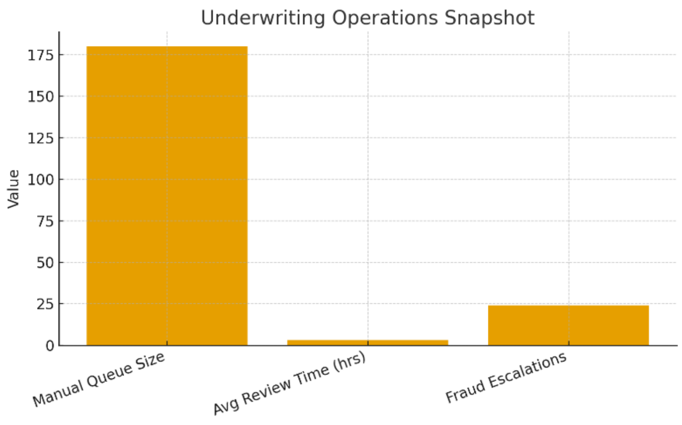

Loan Application – Confirmation
Application Submitted!
Your Reference Number:
Estimated Decision Time: 1 hour
Back to Dashboard

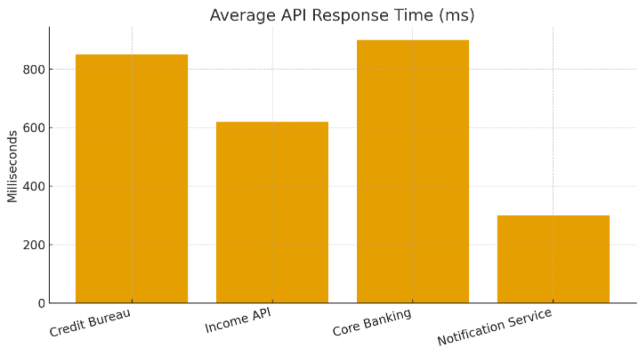
Tools I can use instead of Draw.io: **Figma**

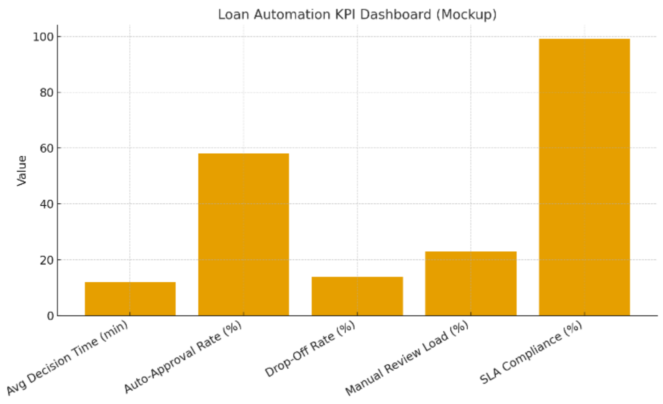
KPI Dashboard Mockup (Power BI / Tableau)

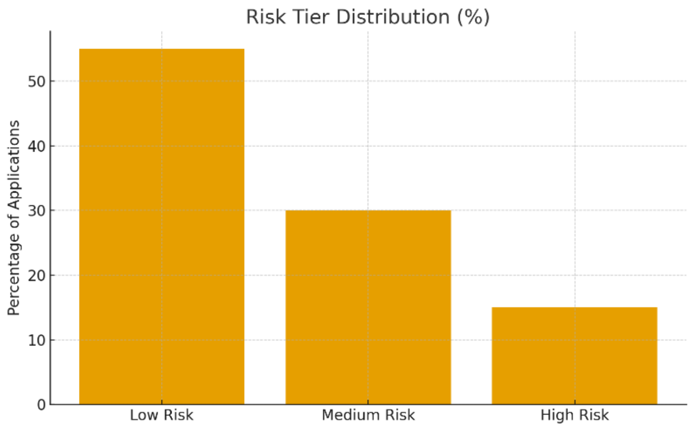
A dashboard illustrating live monitoring of loan automation performance: - Average decision time - Auto-approval rate - Applications per day - Risk distribution - Manual review workload - SLA compliance- etc.



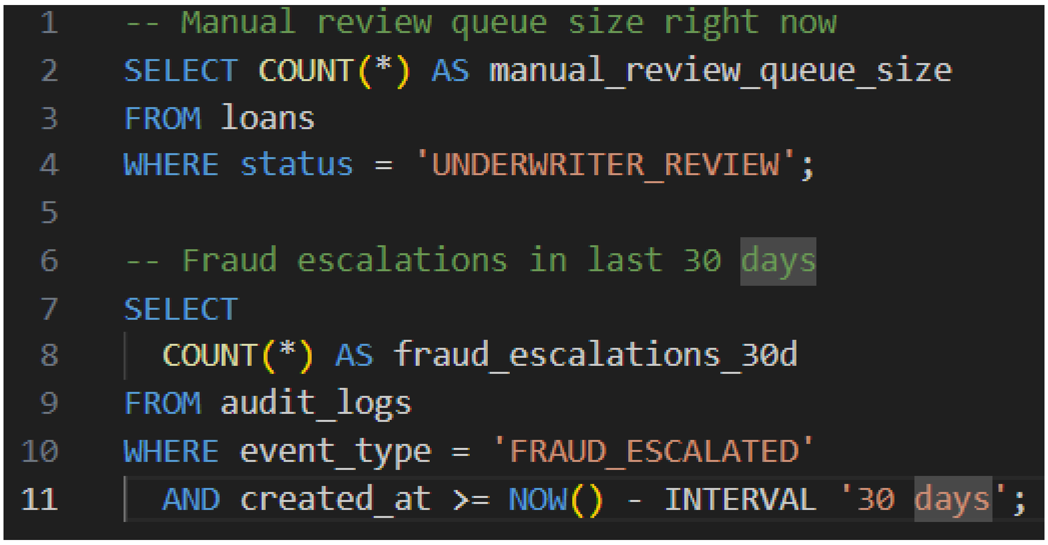

Underwriting Operations Snapshot
Value
175
150
125
100
75
50
25
0
Manual Queue Size
Avg Review Time (hrs)
Fraud Escalations


Average API Response Time (ms)
Milliseconds
800
600
400
200
0
Credit Bureau
Income API
Core Banking
Notification Service


Loan Automation KPI Dashboard (Mockup)
Value
100
80
60
40
20
0
Avg Decision Time (min)
Auto-Approval Rate (%)
Drop-Off Rate (%)
Manual Review Load (%)
SLA Compliance (%)


Risk Tier Distribution (%)
Percentage of Applications
50
40
30
20
10
0
Low Risk
Medium Risk
High Risk

Sample queries used to extract data:



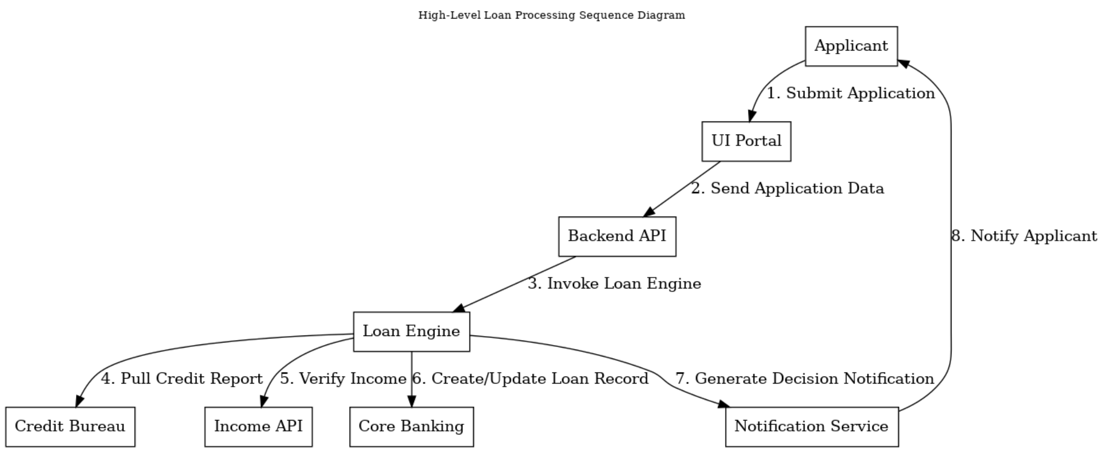
```sql
-- Manual review queue size right now
SELECT COUNT(*) AS manual_review_queue_size
FROM loans
WHERE status = 'UNDERWRITER_REVIEW';

-- Fraud escalations in last 30 days
SELECT
  COUNT(*) AS fraud_escalations_30d
FROM audit_logs
WHERE event_type = 'FRAUD_ESCALATED'
  AND created_at >= NOW() - INTERVAL '30 days';
```

Tools I can use instead of Power BI: **Tableau**

Sequence Diagram (Lucidchart)

Shows the exact order of calls: User → UI → Backend API → Loan Engine → Credit Bureau → Loan Engine → Core Banking → UI.



Tools I can use instead of Draw.io: **Lucidchart**
