Software Requirements Specification

for

AI-Powered Personal Finance Assistant with Blockchain Integration

Version 1.0

Prepared by

Mohammad Owais	23BCE1746	mohammaad.owais2023@vitstudent.ac.in

Instructor:	Nancy Parkavi K
Course:	Software Engineering
Lab Section:	*L3 - L4*
Date:	28-07-2025

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Mohammad Owais	Initial SRS document for AI-Powered Personal Finance Assistant.	28-07-2025

1 Introduction

This Software Requirements Specification (SRS) document defines the requirements for an AI-Powered Personal Finance Assistant with Blockchain Integration. The system is designed to help individuals manage their finances through intelligent automation, personalized recommendations, and secure transaction verification using cutting-edge artificial intelligence and blockchain technologies.

1.1 Document Purpose

This document specifies the software requirements for the AI-Powered Personal Finance Assistant with Blockchain Integration, version 1.0. The SRS covers the complete system that will be developed using the Incremental Development Model over a 10-week period. This specification serves as the foundation for system design, development, testing, and validation activities.

The document establishes a clear understanding between stakeholders regarding system functionality, performance requirements, and technical constraints while providing a comprehensive reference for the development team throughout the project lifecycle.

1.2 Product Scope

The AI-Powered Personal Finance Assistant is a web-based application that revolutionizes personal financial management through intelligent automation and enhanced security. The software provides users with AI-driven transaction categorization, automated budget recommendations, blockchain-based transaction verification, fraud detection capabilities, and an interactive dashboard with comprehensive financial insights.

Key Benefits:

- **Intelligent Automation:** Reduces manual financial data entry through AI-powered categorization
- **Enhanced Security:** Provides immutable transaction records through blockchain integration
- **Personalized Insights:** Delivers tailored financial recommendations based on spending patterns
- **Fraud Protection:** Implements real-time anomaly detection for suspicious activities
- **Financial Education:** Offers proactive guidance for improved financial decision-making

1.3 Intended Audience and Document Overview

This document is intended for client and professor:

- **Development Team:** Complete technical specifications for system implementation
- **Project Instructors:** Academic evaluation criteria and project scope validation
- **System Testers:** Functional and non-functional requirements for test case development
- **Academic Evaluators:** Comprehensive SDLC documentation for coursework assessment

Document Organization:

- **Section 2:** Provides system overview and high-level functionality
- **Section 3:** Details specific functional requirements and use cases
- **Section 4:** Specifies non-functional requirements including performance and security
- **Section 5:** Additional requirements and constraints

1.4 Definitions, Acronyms and Abbreviations

Term	**Definition**
AI	Artificial Intelligence
API	Application Programming Interface
CSV	Comma-Separated Values
ERD	Entity-Relationship Diagram
ML	Machine Learning
ORM	Object-Relational Mapping
SRS	Software Requirements Specification
UML	Unified Modeling Language
Web3	Web3.py Python library for blockchain interaction
Ganache	Local blockchain development environment

1.5 Document Conventions

This document follows IEEE 830-1998 standards for software requirements specifications. Text formatting conventions include:

- **Bold text:** For emphasis and important terms
- *Italic text:* For comments and notes
- Bulleted lists for feature collections

- Table for various tabular information

1.6 References and Acknowledgments

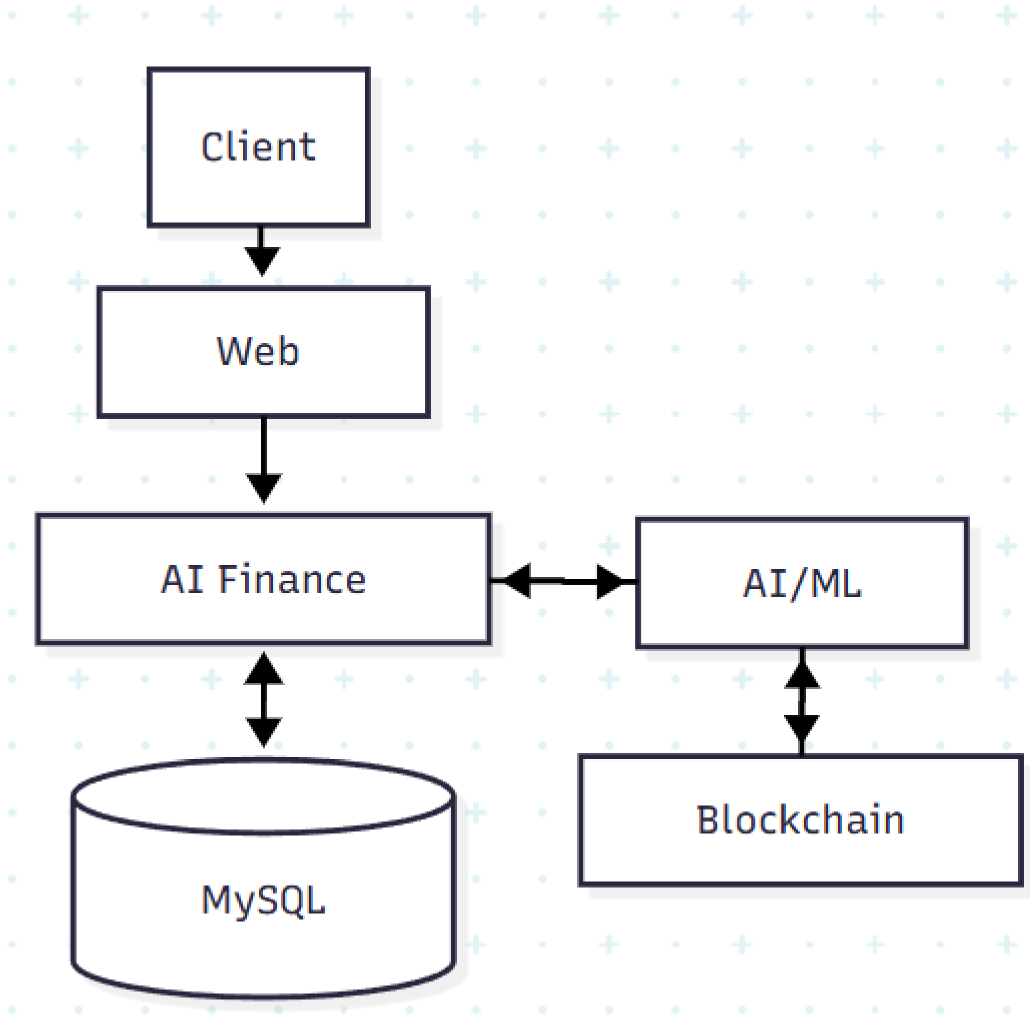
- IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
- Flask Documentation: Official Flask Framework Documentation
- Web3.py Documentation: Ethereum Python Library
- scikit-learn Documentation: Machine Learning Library
- MySQL Documentation: Database Management System

2 Overall Description

2.1 Product Overview

The AI-Powered Personal Finance Assistant with Blockchain Integration is a comprehensive web-based financial management solution designed for individuals seeking intelligent, secure, and automated personal finance management. The system represents a new generation of financial tools that combines artificial intelligence capabilities with blockchain security.

The application operates as a centralized platform where users can import financial data, receive AI-powered analysis and recommendations, and maintain immutable records through blockchain integration.



System Context Diagram

2.2 Product Functionality

The major functions include:

- User authentication and profile management
- Transaction data import and management via CSV uploads
- AI-powered transaction categorization using machine learning
- Budget creation and monitoring with AI recommendations
- Financial analytics and spending pattern insights
- Fraud detection through anomaly detection algorithms
- Blockchain transaction logging for immutable audit trails
- Interactive dashboard with visual data representation
- Goal setting and progress tracking
- Educational content delivery and proactive alerts

2.3 Design and Implementation Constraints

Technical Constraints:

- Development Framework: Flask (Python) for backend development
- Database System: MySQL for production environments
- Frontend Technologies: HTML5, CSS3, JavaScript with Bootstrap
- AI/ML Implementation: scikit-learn, pandas, numpy for machine learning
- Blockchain Integration: Web3.py for Ethereum blockchain interaction
- Development Timeline: 10-week incremental implementation

Academic Constraints:

- Must follow Incremental Development Model methodology
- Complete SDLC documentation required for each increment
- Regular milestone deliverables for academic evaluation

2.4 Assumptions and Dependencies

Key Assumptions:

- Users have basic computer literacy and web browser access
- Users will provide accurate financial data through CSV uploads
- Internet connectivity available for blockchain interactions
- Users understand basic financial management concepts

Major Dependencies:

- External Libraries: Availability of scikit-learn, Web3.py, and Flask frameworks
- Blockchain Network: Reliable access to Ethereum testnet or Ganache environment
- Database System: MySQL server availability and performance
- CSV Data Format: Standardized transaction data format for processing

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The system provides a responsive web-based interface accessible through standard web browsers.

Main Interface Components:

- Login/Registration Page: Secure authentication interface
- Dashboard: Financial overview with summary cards and charts
- Transaction Management: Table-based view with filtering capabilities
- Budget Planning: Interactive budget creation and monitoring
- Analytics Section: Chart-based spending pattern visualizations
- Settings Page: User profile and system configuration

3.1.2 Hardware Interfaces

Client-Side Requirements:

- Standard computer or mobile device with web browser
- Internet connection for system access
- Minimum 2GB RAM for optimal performance

Server-Side Requirements:

- Web server capable of running Flask applications
- Database server for MySQL hosting
- Network connectivity for blockchain communication

3.1.3 Software Interfaces

Database Interface:

- MySQL Database: Primary data storage
- SQLAlchemy ORM: Database abstraction layer
- TCP/IP connection with SSL encryption

Blockchain Interface:

- Ethereum Network: Smart contract deployment
- Web3.py Library: Python interface for blockchain communication
- Ganache: Local blockchain environment for development

3.2 Functional Requirements

3.2.1 FR1: User Authentication and Management

The system shall provide secure user registration, authentication, and profile management.
Sub-requirements:

- FR1.1: Users shall register with email, password, and profile information
- FR1.2: System shall validate email addresses through verification
- FR1.3: Users shall authenticate using email and password credentials
- FR1.4: System shall maintain user sessions with 30-minute timeout
- FR1.5: Users shall update profile information and preferences

3.2.2 FR2: Transaction Data Management

The system shall enable users to import, store, and manage financial transaction data.
Sub-requirements:

- FR2.1: Users shall upload transaction data via CSV file format
- FR2.2: System shall validate and parse CSV data for required fields
- FR2.3: Users shall manually add, edit, and delete transactions
- FR2.4: System shall store transaction data with unique identifiers
- FR2.5: Users shall view transaction history with filtering options

3.2.3 FR3: AI-Powered Transaction Categorization

The system shall automatically categorize transactions using machine learning.
Sub-requirements:

- FR3.1: System shall classify transactions into predefined categories
- FR3.2: AI model shall achieve minimum 80% categorization accuracy
- FR3.3: Users shall review and correct AI categorization decisions
- FR3.4: System shall learn from user corrections to improve accuracy
- FR3.5: System shall provide confidence scores for AI decisions

3.2.4 FR4: Budget Management and Recommendations

The system shall provide AI-generated budget recommendations and monitoring.
Sub-requirements:

- FR4.1: System shall analyze spending patterns for budget suggestions
- FR4.2: Users shall set budget limits for expense categories
- FR4.3: System shall track spending against budget limits real-time
- FR4.4: Users shall receive alerts when approaching budget limits
- FR4.5: System shall provide budget optimization recommendations

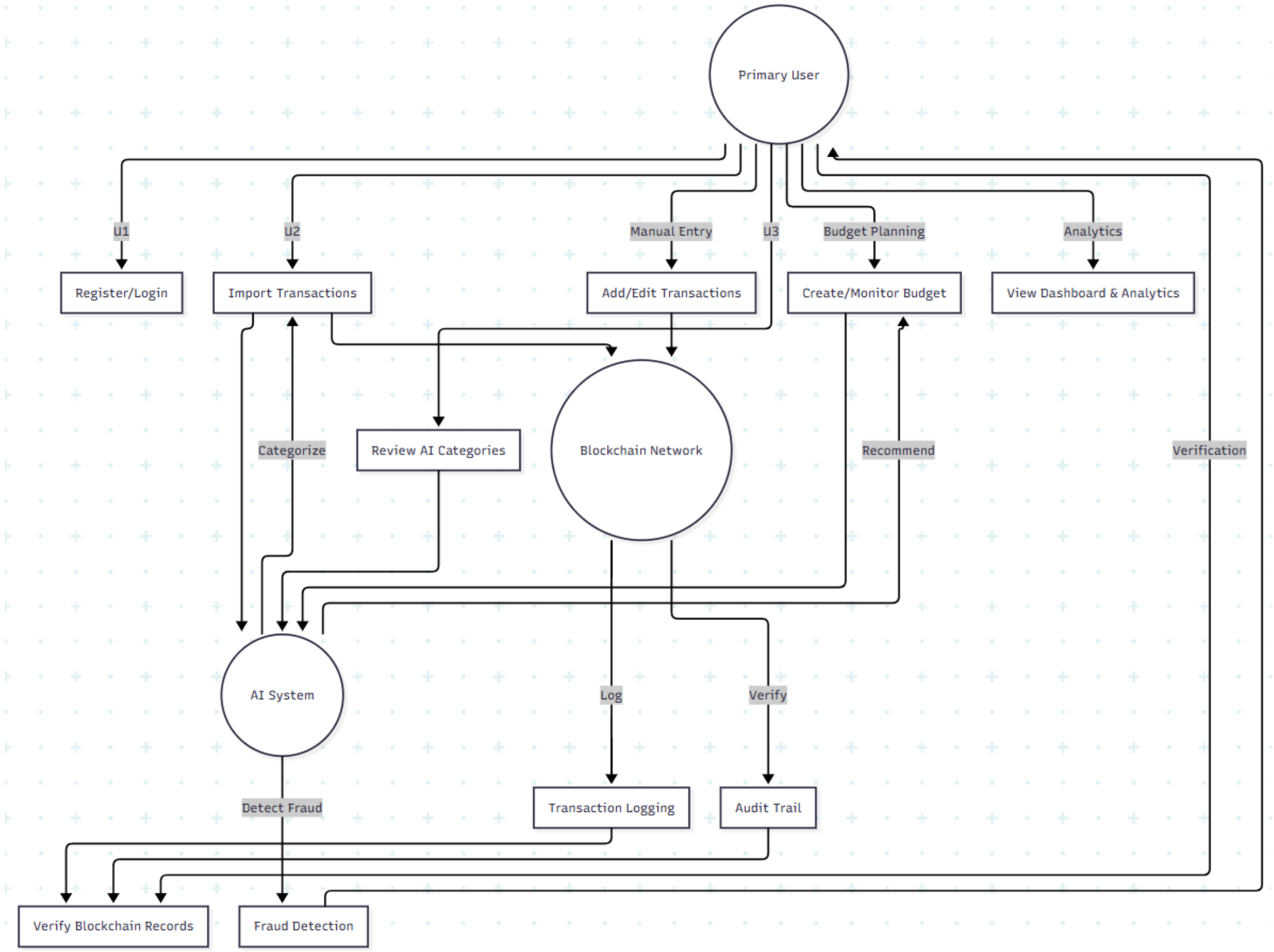
3.2.5 FR5: Blockchain Integration and Transaction Logging

The system shall provide blockchain-based transaction verification.

Sub-requirements:

- FR5.1: System shall log processed transactions to blockchain smart contracts
- FR5.2: Users shall verify transaction authenticity through blockchain records
- FR5.3: System shall maintain 100% reliability for blockchain logging
- FR5.4: Users shall access complete audit trails of financial activities
- FR5.5: System shall provide cryptographic proof of transaction integrity

3.3 Use Case Model



3.3.1 Use Case U1: User Registration and Authentication

Author: Mohammad Owais
Purpose: Enable secure user access to the personal finance system
Requirements Traceability: FR1.1, FR1.2, FR1.3
Priority: High
Preconditions: User has internet access and valid email address
Post-conditions: User account created and authenticated session established

Actors: Primary User

Flow of Events:

1. **Basic Flow:**
   - User navigates to registration page
   - User enters email, password, and profile information
   - System validates input and creates account
   - System sends email verification
   - User confirms email and logs in
2. **Alternative Flow:**
   - User with existing account proceeds to login
3. **Exceptions:**
   - Invalid email format or duplicate account

3.3.2 Use Case U2: Transaction Data Import

Author: Mohammad Owais

Purpose: Allow users to import financial transaction data

Requirements Traceability: FR2.1, FR2.2, FR2.4

Priority: High

Preconditions: User authenticated with CSV transaction data

Post-conditions: Transaction data imported and stored

Actors: Primary User

Flow of Events:

1. **Basic Flow:**
   - User uploads CSV file
   - System validates and parses content
   - System displays transactions for review
   - User confirms import
2. **Exceptions:**
   - Invalid CSV format or missing fields

3.3.3 Use Case U3: AI Transaction Categorization

Author: Mohammad Owais

Purpose: Automatically categorize transactions using AI

Requirements Traceability: FR3.1, FR3.2, FR3.3

Priority: High

Preconditions: Transaction data exists in system

Post-conditions: Transactions categorized with AI predictions

Actors: Primary User, AI System

Flow of Events:

1. **Basic Flow:**
    - AI processes uncategorized transactions
    - System provides categorization suggestions
    - User reviews and corrects as needed
    - System learns from feedback
2. **Exceptions:**
    - AI model fails to categorize transaction

4 Other Non-functional Requirements

4.1 Performance Requirements

P1: Web pages shall load within 2 seconds under normal conditions

P2: AI transaction categorization shall complete within 3 seconds for 100 transactions

P3: Database queries shall return results within 1 second

P4: Blockchain transaction logging shall complete within 30 seconds

P5: System shall support 50 concurrent users without degradation

4.2 Safety and Security Requirements

S1: User passwords shall be encrypted using bcrypt hashing

S2: Database connections shall use SSL/TLS encryption

S3: User sessions shall implement CSRF protection

S4: Financial data shall be encrypted at rest using AES-256

S5: System shall prevent SQL injection through input validation

4.3 Software Quality Attributes

4.3.1 Reliability

R1: System shall maintain 99% uptime during operation
R2: Data integrity maintained through database transactions

4.3.2 Usability

U1: New users shall complete registration within 10 minutes
U2: Interface shall be accessible to users with basic computer literacy

4.3.3 Maintainability

M1: Code architecture shall support easy addition of new features
M2: System shall provide comprehensive logging for debugging

5 Other Requirements

L1: *System shall comply with data privacy regulations for financial information*
D1: *System shall include comprehensive user documentation*
T1: *System shall include built-in help features for user guidance*

Appendix A – Data Dictionary

Data Element	***Description***	***Type***	***Operations***	***Related Requirements***
User	*System user account*	*Entity*	*CRUD*	*FR1.1, FR1.4, FR1.5*
user_id	*Unique user identifier*	*Integer (PK)*	*Generate, Reference*	*All user functions*
email	*User email address*	*String (Unique)*	*Validate, Authenticate*	*FR1.1, FR1.2*
password_hash	*Encrypted password*	*String*	*Hash, Verify*	*FR1.1, S1*
Transaction	*Financial transaction record*	*Entity*	*CRUD*	*FR2.3, FR2.4*
transaction_id	*Unique transaction ID*	*Integer (PK)*	*Generate, Reference*	*FR2.4, FR5.1*
amount	*Transaction amount*	*Decimal(10,2)*	*Calculate, Aggregate*	*FR4.2, FR4.3*
description	*Transaction description*	*String*	*Categorize, Search*	*FR3.1, FR3.2*
category	*Expense category*	*Enum*	*Classify, Group*	*FR3.1, FR4.2*
blockchain_hash	*Blockchain verification hash*	*String*	*Verify, Audit*	*FR5.2, FR5.4*

Appendix B - Group Log

Day 1 (July 26): Foundation

- Reviewed Software Process Model Report and SRS template
- Created Introduction and Overall Description sections
- Established system context and requirements framework

Day 2 (July 27): Core Requirements

- Developed functional requirements FR1-FR5 with sub-requirements
- Created use case model with 3 detailed specifications
- Defined external interfaces and non-functional requirements

Day 3 (July 28): Completion

- Built data dictionary and appendices
- Final document review and formatting
- Prepared 14-page SRS for submission