Software Process Model Report

1. Project Idea

AI-Powered Personal Finance Assistant with Blockchain Integration

A web-based application that helps individuals manage their finances through intelligent automation and secure transaction verification. The system combines:

- AI-powered transaction categorization using machine learning algorithms
- Automated budget recommendations based on spending patterns
- Blockchain-based transaction verification for immutable audit trails
- Fraud detection through anomaly detection algorithms
- Interactive dashboard with financial insights and recommendations

Technology Stack:

- Backend: Flask (Python), SQLAlchemy ORM
- Database: MySQL
- Frontend: HTML, CSS
- AI/ML: scikit-learn, pandas, numpy
- Blockchain: Web3.py, Ethereum/Ganache

2. Software Process Model Used

Incremental Development Model

3. Why This Model Was Chosen

Project Complexity Management

- Multi-technology Integration: The project involves AI/ML, blockchain, web development, and database management. The incremental approach allows focusing on one technology integration at a time.
- Modular Architecture: The application naturally divides into distinct functional modules (authentication, transaction management, AI engine, blockchain layer).

Academic Timeline Alignment

- Short Window: Adapted to fit 4 increments plus final integration within 10 months
- Clear Milestones: Provides concrete deliverables for academic evaluation

- Documentation Requirements: Each increment produces complete SDLC documentation

Risk Mitigation

- Early Risk Detection: Core functionality tested early before adding complex features
- Stable Foundation: Each increment provides a solid base for subsequent development
- Fallback Options: If later increments face challenges, earlier increments remain functional

Resource Considerations

- Small Team Size: Suitable for 1-3 developer teams typical in academic projects
- Learning Curve: Allows time to master new technologies (Web3, ML) progressively
- Quality Assurance: Enables thorough testing and documentation at each stage

4. Implementation

Increment 1: Foundation & Authentication (Weeks 1-3)

Week 1: Requirement Analysis

- Define user authentication and basic system requirements
- Create Software Requirements Specification (SRS)
- Document functional and non-functional requirements
- Establish system architecture framework

Week 2: Design and Development

- Design system architecture and database ERD
- Develop Flask application structure with user authentication
- Create basic HTML templates and CSS styling
- Set up MySQL/SQLite database
- Implement core authentication functionality

Week 3: Testing and Implementation

- Unit testing for authentication functions
- Database connection testing
- User interface and security testing

- Deploy functional authentication system
- Demonstrate working login system

Increment 2: Transaction Management (Weeks 4-5)

Week 4: Requirement Analysis and Development

- Define transaction data requirements and CSV upload specifications
- Create use case diagrams for transaction management
- Design transaction database schema

Week 5: Testing and Implementation

- Complete CRUD operations for transactions
- Implement CSV upload functionality
- Create transaction listing interfaces
- Test transaction operations and CSV processing
- Integration testing with authentication
- Deploy transaction management system

Increment 3: AI Intelligence Engine (Weeks 6-7)

Week 6: Requirement Analysis and Development

- Define AI model requirements for categorization
- Specify budget prediction and fraud detection needs
- Begin ML model development for transaction categorization
- Create behavioral modeling diagrams

Week 7: Testing and Implementation

- Complete AI model development and training
- Implement budget prediction algorithms
- Create AI-powered spending analysis features
- Test AI model accuracy with sample data
- Deploy AI engine integrated with transaction system
- Performance optimization and demonstration

Increment 4: Blockchain Integration (Weeks 8-9)

Week 8: Requirement Analysis and Development

- Define blockchain integration and smart contract requirements
- Document security and audit trail specifications
- Develop smart contracts for transaction logging
- Begin Web3.py integration

Week 9: Testing and Implementation

- Complete blockchain verification interface
- Set up Ganache for testing
- Test smart contract functionality
- Validate blockchain transaction logging
- Security testing for blockchain components
- Deploy blockchain-integrated system
- Complete system integration testing

Final Integration (Week 10)

Week 10: System Integration and Final Deployment

- Combined Analysis, Development, Testing, and Implementation:
    - Review all requirements for completeness
    - Final system integration and UI refinements
    - End-to-end system testing
    - User acceptance testing
    - Security vulnerability assessment
    - Final deployment and optimization
    - Complete documentation package
    - Project demonstration and presentation

5. Conclusion

In conclusion, the Incremental Development Model provides a structured and effective framework for the development of the AI-Powered Personal Finance Assistant with

Blockchain Integration. This model enables systematic management of complex technological components by breaking down the project into manageable increments, ensuring early delivery of functional software, and supporting comprehensive documentation through each phase of the project.

The incremental approach proves particularly valuable for this project due to its ability to accommodate the learning curve associated with emerging technologies like AI and blockchain while maintaining steady development progress. Each increment builds upon previous functionality, creating a stable foundation that reduces integration risks and provides fallback options if later components encounter development challenges.