Software Requirements Specification

For

SCAM DETECTION FOR ONLINE TRANSCACTIONS

Version 1.0 approved

Prepared by

1. **Gayathri.S(23MID0192)**

<VIT>

Table of Contents

Revision History

Date	Version	Description	Author
	1		

1. Introduction

1.1 Purpose

The purpose of this project is to enhance the security of online transactions by detecting and preventing scams in real time. It aims to minimize financial losses, build user trust, and support regulatory compliance through advanced technologies like machine learning and anomaly detection. The system provides easy scam reporting, educates users about fraud, and continuously improves to stay ahead of emerging threats, ensuring safer and more reliable digital payments.

1.2 Intended Audience and Reading Suggestions

1. **Intended Audience:**
   The primary audience for this document includes:

   1.Developers and Engineers: To understand the technical details, system architecture, and implementation specifics of the scam detection project.

   2.Project Stakeholders: Business owners, financial institutions, and decision-makers who need insights into the project's objectives, outcomes, and benefits.

   3.Quality Assurance Teams: To comprehend the testing strategies, use cases, and metrics used to validate the system.

   4.End-Users: Individuals or organizations using the system to learn about its functionality, features, and user interface.

   5.Researchers and Academics: To explore the methodologies, algorithms, and technologies implemented for scam detection.

2. **Reading Suggestions**
   To ensure effective use of this document, follow these recommendations:

   1.Overview: Begin with the Introduction section to understand the project's purpose and objectives.

   2.Technical Details: Developers should focus on the System Design and Implementation sections for architecture, tools, and algorithms.
   Quality assurance teams should study the *Testing* section for test cases and validation metrics.

   3.User Guidance: End-users should review the User Interface and Usage Instructions sections for practical steps to use the system.

4.Key Takeaways: Project stakeholders should focus on the Benefits, Impact, and Conclusion sections to assess the value of the project.

5.Further Exploration: Researchers and academics should delve into the References and Appendices for additional resources and in-depth explanations of methodologies.

1.3 Project Scope

- **User Authentication & Verification:**
  1. Multi-Factor Authentication (OTP, biometrics).
  2. Device fingerprinting & IP tracking.
  3. CAPTCHA to block bots.

- **Real-Time Fraud Detection:**
  1. AI-Based Risk Scoring (Machine learning detects anomalies).
  2. Rule-Based Checks (Flagging large/unusual transactions).
  3. Velocity Checks (Multiple transactions in a short time).
  4. Blacklist/Whitelist (Block known fraudsters, allow trusted users).

- **Secure Payment Processing:**
  1. 3D Secure (3DS), CVV, AVS Checks* for card verification.
  2. AI-powered monitoring of payment patterns.
  3. Chargeback & refund fraud detection.

- **AI & Anomaly Detection:**
  1. Detect outliers using *machine learning
  2. Graph-based fraud detection* for connected fraudsters.
  3. Deep learning for pattern recognition.

- **Alerts & Reporting:**
  1. Instant SMS/Email alerts for suspicious activity.
  2. Admin dashboard for real-time fraud tracking.

- **Compliance & Security:**
  1. KYC & AML checks.
  2. PCI DSS compliance for payment security.

- **Tech Stack:**
  1. Backend: Python, Java, Node.js
  2. ML: TensorFlow, Scikit-Learn
  3. Database: PostgreSQL, MongoDB, Neo4j

1.4 References

1. Karthikeyan, T., Govindarajan, M., & Vijayakumar, V. (2023). An effective fraud detection using competitive swarm optimization based deep neural network. *Measurement Sensors*, *27*, 100793. https://doi.org/10.1016/j.measen.2023.100793

2. Montague, D. A. (2010). Essentials of Online payment Security and Fraud Prevention. John Wiley & Sons.

2. Overall Description

2.1 Process Perspective

From a process perspective, verifying whether an online platform is secure for payments involves a structured, step-by-step approach. Here's a breakdown:

Process Perspective for Payment Security Verification

1. Initiation & Planning

- Define security requirements (e.g., PCI DSS compliance, SSL, fraud prevention).
- Identify target platform(s) for evaluation.
- Select appropriate security assessment tools and testing methods.
- Establish key metrics and benchmarks (e.g., SSL grade, API security score).

2. Security Assessment & Testing

A. Network & Data Encryption Check

- Verify SSL/TLS encryption (use tools like SSL Labs to check certificate validity).
- Ensure HTTPS enforcement and no mixed content issues.
- Test for HSTS (HTTP Strict Transport Security) implementation.

B. Payment Gateway & Transaction Security

- Identify the payment gateway provider and check for PCI DSS compliance.
- Test for secure checkout flow (no sensitive data exposure in URLs).
- Check for tokenization and encryption in payment processing.

C. Authentication & Access Control

- Verify the presence of Multi-Factor Authentication (MFA) for user accounts.
- Check password security policies (e.g., minimum length, complexity, expiry).
- Test role-based access control (RBAC) for payments and user permissions.

D. Fraud Prevention & Anomaly Detection

- Assess whether the platform has AI-driven fraud detection.
- Verify CVV and Address Verification System (AVS) checks.
- Ensure protection against chargeback fraud.

E. API & Third-Party Integration Security

- Scan for insecure API endpoints (use tools like Postman, Burp Suite).
- Verify OAuth 2.0 or OpenID Connect for authentication.
- Ensure rate limiting and input validation to prevent abuse.

F. Compliance & Legal Considerations

- Check compliance with GDPR, CCPA, or regional data protection laws.
- Ensure a clear and enforceable privacy policy is in place.
- Verify if users are informed of security policies (e.g., terms of service).

3. Analysis & Risk Assessment

- Compile security findings and prioritize risks based on severity.
- Compare results against industry best practices.
- Assign a risk score to each issue (e.g., High, Medium, Low).

4. Reporting & Recommendations

- Create a detailed security report with identified vulnerabilities.
- Provide remediation steps for fixing security gaps.
- Suggest continuous monitoring strategies for long-term security.

5. Continuous Monitoring & Improvement

- Implement real-time threat detection systems.
- Conduct periodic security audits and penetration testing.
- Ensure the platform follows best practices for secure payments (e.g., updated libraries, patching vulnerabilities).

2.2 Process Features

To check whether an online platform is secure for payments, you can follow a structured process with key features:

1. SSL/TLS Security

- Check if the platform uses HTTPS (SSL/TLS encryption) to protect data.
- Verify the SSL certificate (e.g., using online tools like SSL Labs).

2. Payment Gateway Security

- Ensure the platform uses a reputable payment gateway (e.g., Stripe, PayPal).
- Check for PCI DSS compliance (Payment Card Industry Data Security Standard).

3. Authentication & Authorization

- Verify Multi-Factor Authentication (MFA) for user accounts.
- Ensure role-based access control (RBAC) for handling payments.

4. Secure Data Handling

- Look for tokenization of payment data.
- Ensure encryption of sensitive user information.

5. Fraud Prevention Measures

- Check for AI-based fraud detection mechanisms.
- Look for CVV verification and address verification systems (AVS).

6. User Privacy & Compliance

- Ensure GDPR, CCPA, or other regional compliance.
- Check if the platform has a clear privacy policy.

7. Platform Reputation & Reviews

- Analyse user reviews on security issues.
- Verify if the platform has been breached in the past.

8. Secure APIs & Integrations

- Ensure OAuth 2.0 or OpenID Connect for authentication.
- Validate API security with proper authentication headers.

9. Regular Security Audits

- Look for security certifications (e.g., ISO 27001).
- Check if the platform undergoes penetration testing.

10. Customer Support & Dispute Resolution

- Ensure dispute resolution mechanisms exist.
- Look for secure refund policies.

2.3 User Classes and Characteristics

To analyse user classes and characteristics for a secure online payment verification process, we can categorize users based on their roles and responsibilities in the system.

User Classes & Their Characteristics

1. End Users (Customers)
Characteristics:

- Non-technical users making online transactions.
- Require a secure and seamless payment experience.
- Concerned with fraud prevention, privacy, and ease of use.
- Expect MFA, encrypted transactions, and refund policies.

2. Platform Owners/Admins
Characteristics:

- Manage platform security, transactions, and user data.
- Ensure compliance with PCI DSS, GDPR, CCPA.
- Implement security policies (e.g., role-based access, fraud prevention).
- Perform regular audits, risk assessments, and penetration testing.

3. Developers & Security Engineers
Characteristics:

- Technical experts responsible for platform security.
- Implement SSL/TLS, tokenization, encryption, API security.
- Set up firewalls, fraud detection, and real-time monitoring.
- Conduct penetration testing and vulnerability assessments.

4. Payment Gateway Providers (Third Parties)
Characteristics:

- Provide secure transaction processing services.
- Must comply with PCI DSS and other security standards.
- Implement fraud detection, tokenization, and chargeback prevention.
- Ensure uptime, scalability, and encrypted data storage.

5. Compliance Officers & Auditors
Characteristics:

- Ensure the platform follows legal and regulatory requirements.
- Perform risk assessments and compliance checks.
- Validate user data protection and privacy measures.
- Review security policies and incident response plans.

6. Hackers & Malicious Actors (Threat Agents)
Characteristics:

- Attempt phishing, identity theft, fraud, and payment interception.
- Exploit weak SSL/TLS configurations, API vulnerabilities, and weak authentication.
- Use bots and automated attacks to test security gaps.
- Target stolen credentials, insecure payment flows, and outdated software.

2.4 Operating Environment

Operating Environment for Secure Payment Verification

The operating environment defines the technical, regulatory, and functional conditions under which the secure payment verification process operates. It includes hardware, software, networks, legal requirements, and external factors that impact the security of online transactions.

1. Technical Environment

This covers the infrastructure, platforms, and tools used for payment processing and security verification.

A. Web & Mobile Platforms

- E-commerce websites, mobile apps, and digital wallets where transactions occur.
- Use of progressive web apps (PWA) and native apps for online payments.
- Browser and OS security dependencies (e.g., Chrome, iOS, Android security settings).

B. Networking & Communication

- Internet-based payment transactions via HTTPS, TLS 1.2/1.3.
- Cloud-based CDN and WAF (Web Application Firewalls) for DDoS protection.
- VPN and zero-trust architectures for secure remote access.

C. Server & Database Security

- Cloud and on-premise servers for handling transactions (AWS, Azure, Google Cloud).
- Encryption at rest (AES-256) and encryption in transit (TLS 1.3).
- Secure NoSQL/SQL databases (PostgreSQL, MongoDB, MySQL) for transaction storage.

D. API & Third-Party Integrations

- Secure APIs using OAuth 2.0, JWT, and OpenID Connect.
- Integration with payment gateways (PayPal, Stripe, Razor pay).
- Webhooks for real-time transaction verification.

2. Regulatory & Compliance Environment

Secure payment platforms must adhere to legal and compliance requirements to ensure trust and security.

- PCI DSS (Payment Card Industry Data Security Standard) – Required for handling credit card payments.
- GDPR (General Data Protection Regulation) – Data privacy laws for European users.
- CCPA (California Consumer Privacy Act) – Protects customer data in California.
- PSD2 (Payment Services Directive 2) – Strong Customer Authentication (SCA) requirements in Europe.
- ISO 27001 – Information security management standards.
- AML (Anti-Money Laundering) & KYC (Know Your Customer) – Financial compliance measures.

3. Security & Threat Landscape

This includes risks and countermeasures related to online payments.

A. Security Threats

- Phishing attacks – Social engineering to steal payment credentials.
- Man-in-the-Middle (MITM) attacks – Interception of payment data.
- Carding & Chargeback fraud – Automated bot attacks on payment systems.
- Malware & Ransomware – Infecting systems to steal or encrypt transaction data.

B. Security Countermeasures

- Multi-Factor Authentication (MFA) – To prevent unauthorized access.
- Fraud detection & anomaly detection using AI/ML.
- End-to-end encryption & tokenization for sensitive data.
- Penetration testing & vulnerability scanning to detect weaknesses.

4. Business & Operational Constraints

- Transaction speed vs. security – Must balance user experience and safety.
- Scalability requirements – Handling peak loads securely.
- Cross-border payments – Managing different regulatory requirements.
- User education – Training customers on secure payment practices.

2.5 Design and Implementation Constraints

Here are the Design and Implementation Constraints for your secure online payment verification project:

Design Constraints

These are the restrictions and requirements that influence the architecture, security, and user experience of the payment verification system.

1. Security & Compliance Requirements

- Must comply with PCI DSS, GDPR, PSD2, and CCPA regulations.
- Data encryption (AES-256, TLS 1.3) for protecting transactions.
- MFA (Multi-Factor Authentication) to prevent unauthorized access.
- Implement role-based access control (RBAC) to restrict payment-related functions.
- Adhere to Open Web Application Security Project (OWASP) security best practices.

2. Payment Gateway & API Limitations

- The system depends on third-party payment gateways (e.g., PayPal, Stripe, Razorpay), which may impose limitations on API calls, security policies, and transaction fees.
- Limited API rate limits & transaction processing speed based on the provider's infrastructure.
- Integration with OAuth 2.0, JWT authentication, and secure API endpoints.

3. Performance & Latency Constraints

- Payment processing must be fast and reliable (<2 seconds per transaction).
- Fraud detection algorithms should work in real-time without delaying transactions.
- The system should be scalable to handle high traffic during peak hours (e.g., Black Friday sales).

4. User Experience (UX) vs. Security Trade-offs

- High security (MFA, CAPTCHA) must be balanced with a smooth user experience to prevent cart abandonment.
- The platform must be responsive and accessible on mobile and web devices.
- Compliance with ADA (Accessibility) guidelines to ensure usability for all users.

Implementation Constraints

These are restrictions related to the development, deployment, and maintenance of the system.

1. Technology Stack & Infrastructure

- Must support multi-platform compatibility (Web, iOS, Android).
- Use secure backend frameworks such as Spring Security (Java), Django (Python), or Node.js (JavaScript).
- Must decide between cloud-based hosting (AWS, Azure, GCP) or on-premise deployment.
- Database constraints – Choosing between relational (PostgreSQL, MySQL) or NoSQL (MongoDB, Firebase) databases for storing transaction data.

2. Integration with External Services

- The platform needs to integrate with banking APIs for verification.
- Secure API communication using TLS 1.3 and mutual authentication.
- Dependency on fraud detection and machine learning models, which require real-time data processing.

3. Regulatory & Legal Compliance

- Transaction data storage must comply with data localization laws in different countries.
- Must implement user consent mechanisms for data collection and processing.
- Adherence to AML (Anti-Money Laundering) and KYC (Know Your Customer) requirements for fraud detection.

4. Cost & Resource Constraints

- Payment security measures (e.g., encryption, fraud detection AI) require high computational power and may increase infrastructure costs.
- Regular security audits and penetration testing require additional budget and resources.
- Transaction fees imposed by payment gateways can impact the platform's profitability.

2.6 Assumptions and Dependencies

These are the **assumptions** made about the system and its environment and the **dependencies** on external factors that may affect implementation and performance.

1. Assumptions

These are the conditions that are expected to be true for the system to function properly.

A. Security & Compliance Assumptions

- The payment platform complies with industry security standards such as PCI DSS, GDPR, PSD2, and CCPA.
- All transactions are encrypted using TLS 1.3 and AES-256 encryption.
- Users follow best security practices, such as not sharing credentials and enabling MFA.
- Payment gateways and third-party providers maintain their security protocols.

B. Technical Assumptions

- The platform will use a secure API for communication between services.
- Payment transactions will be processed in real time with minimal delay.
- The system will have fraud detection mechanisms that can detect and prevent fraudulent transactions.
- The backend infrastructure (cloud or on-premises) will be available and scalable to handle peak loads.

C. User Behaviour Assumptions

- Users will enter valid payment details and follow standard checkout procedures.
- Customers will update their payment methods when required (e.g., expired credit cards).
- Users will be aware of phishing and fraud risks and take necessary precautions.

D. Regulatory Assumptions

- Payment processing laws and security standards will remain stable without drastic changes.
- The platform will have legal support to handle compliance issues.
- Data protection regulations will not significantly impact API integrations.

2. Dependencies

These are external factors that the payment security system **relies on** for proper functioning.

A. Third-Party Payment Gateways & APIs

- The system depends on payment processors (e.g., PayPal, Stripe, Razorpay) for transactions.
- API downtime or rate limits from third-party providers may affect transactions.
- Compliance with OAuth 2.0, JWT, and secure API authentication from external services

.

B. Network & Infrastructure

- Secure payment processing depends on stable internet connections.
- Availability of cloud services (AWS, Azure, GCP) for hosting and scaling.
- Web Application Firewalls (WAF) and CDNs (Cloudflare, Akamai) for security and performance.

C. Device & Browser Compatibility

- Payments rely on secure browser versions that support TLS 1.3 and HTTPS.
- Mobile payments depend on iOS and Android compatibility.
- Older devices may not support advanced encryption algorithms.

D. Fraud Detection & Machine Learning Models

- Fraud detection systems rely on real-time transaction data.
- Machine learning models must be trained with updated datasets.
- Dependence on third-party fraud detection services (e.g., Riskified, Forter) for AI-driven security.

E. Compliance & Legal Updates

- The system depends on financial regulators and government agencies for legal compliance.
- Any changes in AML (Anti-Money Laundering) or KYC (Know Your Customer) laws may require updates.
- GDPR and data protection laws may affect how user data is stored and processed.

3. System Features

3.1 User Authentication

User authentication ensures that only authorized users can access and perform payment transactions securely. This feature implements multi-layered security to prevent unauthorized access and fraud.

1. Functional Requirements

- **Secure Login Methods**
  - Users can log in using email & password, phone number & OTP, or biometric authentication.
  - The system enforces password complexity rules (e.g., minimum 8 characters, special symbols, uppercase/lowercase letters).
  - Session management to prevent unauthorized access.

- **Multi-Factor Authentication (MFA)**
  - MFA is mandatory for high-risk transactions (e.g., large payments, adding a new payment method).
  - Supports SMS OTP, email OTP, authenticator apps (Google Authenticator, Authy), and biometrics (Face ID, fingerprint).
  - Adaptive authentication based on user behavior (e.g., unusual IP or device triggers MFA).

- **Role-Based Access Control (RBAC)**
  - Different access levels for users, merchants, and admins.
  - Admins have control over user permissions and security policies.

- **Single Sign-On (SSO) & Social Logins *(Optional)***
  - Users can authenticate via Google, Apple, Facebook, or Microsoft accounts.
  - SSO implementation for enterprise and business accounts.

- **Logout & Session Management**
  - Auto-logout after a period of inactivity (e.g., 10 minutes for sensitive operations).
  - Secure session handling with refresh tokens & short-lived access tokens.

2. Security Features

- **Encryption & Secure Data Storage**
  - Passwords are stored using bcrypt (hashed & salted).
  - Sensitive authentication data is never stored in plain text.

- **Rate Limiting & Bot Protection**
  - Failed login attempts are limited (e.g., 5 attempts per 10 minutes) to prevent brute-force attacks.
  - reCAPTCHA or hCaptcha integration for bot protection.

- **Device & Location-Based Authentication**
  - If login is attempted from a new device or location, an additional verification step is required.
  - Users receive an alert email or push notification for logins from unknown devices.

- **Audit Logs & Monitoring**
  - All authentication attempts are logged and monitored for unusual activities.
  - Alerts for suspicious login attempts (e.g., too many failed attempts, login from unusual locations/IPs).

3. User Experience (UX) Considerations

- **Fast & Seamless Authentication**
  - Login should take <2 seconds under normal conditions.
  - Passwordless login options (e.g., OTP-based login) for a smooth experience.

- **Recovery Options**
  - Users can reset their password via email or phone verification.
  - Account recovery flow for cases where users lose access to their MFA device.

- **Customizable Security Settings**
  - Users can enable/disable MFA, change passwords, and manage trusted devices.
  - The system provides recommendations for stronger security settings.

3.2 Payment Management:

Payment management ensures that transactions are processed securely, accurately, and efficiently while maintaining user trust.

1. Secure Payment Processing

- Supports multiple payment methods (credit/debit cards, digital wallets, UPI, bank transfers).
- Uses end-to-end encryption (TLS 1.3, AES-256) to secure payment data.
- Tokenization to replace card details with a secure token to prevent fraud.

2. Payment Authorization & Verification:

- Two-Factor Authentication (2FA) for high-value transactions.
- 3D Secure (3DS) authentication for card transactions (Verified by Visa, Mastercard Secure Code).
- AI-based fraud detection to analyze user behavior and prevent unauthorized transactions.

3. Real-Time Transaction Processing:

- Instant validation of payment details and fund availability.
- Automated handling of payment failures, retries, and chargebacks.
- Supports delayed payments or scheduled transactions.

4. Refund & Dispute Management:

- Users can request refunds through their dashboard.
- Automated chargeback handling for disputed transactions.
- Transparent tracking of refund status and resolution timelines.

5. Transaction History & Reports:

- Users can view detailed payment history with date, amount, and status.
- Generate monthly statements and export transaction data (PDF, CSV).
- Categorization of payments (e.g., bills, subscriptions, purchases).

6. Subscription & Recurring Payments:

- Supports automatic billing for subscriptions and memberships.
- Allows users to modify, pause, or cancel subscriptions.
- Sends reminders for upcoming payments.

7. Payment Notifications & Alerts:

- Real-time SMS/email notifications for successful, failed, or suspicious transactions.
- Push notifications for mobile users.
- Alerts for low balance, expired cards, and due payments**.**

8. Currency & International Payment Support:

- Multi-currency support with real-time exchange rate conversion.
- Compliance with international regulations (PSD2, GDPR, AML, KYC).

4. External Interface Requirements

4.1 User Interfaces

The User Interface (UI) plays a crucial role in ensuring a seamless and secure payment experience. The system will have intuitive, responsive, and user-friendly interfaces for different users.

1. Web Interface (Desktop & Mobile Browsers):

- Dashboard: Displays recent transactions, account balance, and payment history.
- Payment Page: Secure and simple form for entering payment details.
- Transaction Status: Real-time updates on successful, pending, or failed payments.
- User Profile & Settings: Allows users to update payment methods, security settings, and notification preferences.

➢ **Design Considerations:**
1. Responsive design (compatible with desktops, tablets, and smartphones).
2. Dark & light mode support.
3. Multi-language support for international users.

2. Mobile Application Interface (iOS & Android) :

- Biometric authentication (Face ID, fingerprint) for quick and secure access.
- Push notifications for transaction alerts and security warnings.
- QR Code Payments for contactless transactions.
- One-tap payments for frequently used payment methods.

➢ **Design Considerations:**
1. Native UI/UX using Material Design (Android) and Human Interface Guidelines (iOS).
2. Offline mode for storing recent transaction data.
3. Gesture-based navigation for ease of use.

3. Admin Panel (For System Administrators & Merchants):

- User & Merchant Management: View and manage accounts, transactions, and security settings.
- Fraud Detection Dashboard: Displays suspicious activity and risk analysis.
- Reports & Analytics: Insights into transaction trends, failed payments, and refunds.
- System Logs: Monitor authentication attempts, payment failures, and chargeback disputes.

➢ **Design Considerations:**
1. Role-based access control (RBAC)* for different admin privileges.
2. Graphical reports & charts* for analytics.

3. Secure login with MFA for administrators.

4. Accessibility Features:

- Screen reader compatibility (for visually impaired users).
- Keyboard shortcuts for navigation.
- Voice commands for hands-free transactions.

4.2 Software Interfaces

The system will integrate with *third-party payment gateways* to process transactions securely. This ensures smooth handling of online payments while maintaining compliance with financial regulations.

1. Supported Payment Gateways:

The system will support integration with multiple payment gateways, including:

- Stripe – Best for card payments & subscriptions.
- PayPal – Widely used for digital payments & global transactions.
- Razorpay – Ideal for businesses in India.
- Square – Suitable for both online and POS payments.
- Apple Pay / Google Pay – For seamless mobile transactions.

2. API Integration Approach:

Integration with payment gateways will be done using their RESTful APIs or SDKs.

Each transaction will follow a standard payment flow, including:

a) Payment Initialization:

- The system will generate a secure payment request using API keys.
- It will send payment details (amount, currency, user ID, etc.) to the gateway.

b) Authentication & Authorization:

- The gateway will verify card details using 3D Secure (3DS).
- If required, the user will complete two-factor authentication (OTP, biometrics).

c) Payment Processing & Response Handling:

- Once the gateway processes the payment, it will return a success, failure, or pending status.
- The system will store the transaction ID and update the user's account.

d) Refunds & Chargebacks

- Supports partial and full refunds via API.
- Integrates dispute resolution mechanisms for chargebacks.

3. Security & Compliance Considerations:

- PCI-DSS Compliance – Ensures secure handling of card data.
- Tokenization – Sensitive card details are replaced with a token for added security.
- End-to-end encryption – TLS 1.3 encryption for all transactions.
- Fraud detection – AI-based anomaly detection for suspicious payments.

5. Webhooks for Payment Status Updates:

Webhooks will be used to listen for real-time payment events, such as:

- payment_intent. succeeded → Successful payment.
- payment_intent.payment_failed → Failed payment.
- charge. refunded → Refund processed.

4.3 Communications Interfaces

The system will use *RESTful APIs* to enable secure, efficient, and scalable communication between the client (web/mobile apps) and the server (backend system). These APIs will handle payment transactions, user authentication, and other essential functionalities.

1. API Design Principles

- RESTful architecture – Ensuring stateless communication.
- JSON format – Standardized data exchange format.
- Secure endpoints – Enforced via OAuth 2.0, JWT tokens, and TLS encryption.
- Rate limiting – Prevents abuse and ensures API stability.
- Versioning – Ensures backward compatibility (/api/v1/...).

2. Key API Endpoints

a) User Authentication APIs

- POST /api/v1/auth/register → Registers a new user.
- POST /api/v1/auth/login → Authenticates user and returns a JWT token.
- POST /api/v1/auth/logout → Logs out and revokes session.
- POST /api/v1/auth/refresh → Generates a new access token.

b) Payment Processing APIs

- POST /api/v1/payments/initiate → Creates a new payment request.
- GET /api/v1/payments/status/{transactionId} → Retrieves payment status.
- POST /api/v1/payments/refund/{transactionId} → Processes refunds.

c) Transaction History APIs

- GET /api/v1/transactions/user/{userId} → Fetches user's payment history.
- GET /api/v1/transactions/{transactionId} → Retrieves transaction details.

d) Webhook API (for Payment Gateway Notifications)

- POST /api/v1/webhooks/payment-status → Receives real-time updates from Stripe, PayPal, etc.

4. API Security Measures:

- OAuth 2.0 & JWT Authentication for secure access.
- Role-based access control (RBAC) for restricted endpoints.
- TLS 1.3 encryption for all communications.
- Input validation & sanitization to prevent injection attacks.

5. Other Nonfunctional Requirements

5.1 Performance Requirements:

The system must ensure high performance, scalability, and low-latency transactions to provide a seamless online payment experience**.**

1. Response Time:

- Payment processing latency: Must be ≤ 2 seconds for 95% of transactions.
- API response time: Must be ≤ 300ms under normal load.
- Dashboard loading time: Must not exceed 1.5 seconds for users with up to 100 transactions**.**

2. Scalability:

- Must handle 10,000 concurrent users without degradation.
- Auto-scaling must be enabled to handle peak loads (e.g., Black Friday sales).
- Load balancers will distribute requests across multiple servers.

3. Throughput:

- The system must support 500 transactions per second (TPS) at peak load.
- Payment gateway requests must be processed asynchronously to improve speed.

4. Availability:

- 99.99% uptime with no more than 52 minutes of downtime per year.
- Automatic failover in case of server failure.
- Geo-distributed data centers to minimize regional disruptions.

5. Data Processing & Storage:

- Must support real-time transaction processing.
- Database queries must complete within 200ms.
- Read and write operations should not impact real-time processing.

6. Load Testing & Benchmarking:

- Stress tests must simulate 100,000 concurrent transactions.
- Database indexing must optimize query performance.
- Caching mechanisms (Redis, Memcached) must reduce repeated computations.

5.2 Safety Requirements

The system must ensure the safety and integrity of payment transactions, user data, and financial information to prevent fraud, data breaches, and unauthorized access.

1. Data Security & Protection:

- End-to-End Encryption (E2EE) – All transactions and sensitive user data must be encrypted using AES-256.
- TLS 1.3 Protocol – Secure all communication between client and server.
- Tokenization – Replace sensitive card details with tokens to prevent exposure.

2. Fraud Prevention & Risk Mitigation*

- AI-based fraud detection – Identify suspicious transaction patterns in real time.
- Two-Factor Authentication (2FA) – Mandatory for login and high-value transactions.
- Geo-blocking & IP filtering – Restrict payments from blacklisted regions or unauthorized locations.

3. System Fail-Safe & Recovery:

- Automatic transaction rollback in case of payment failures.
- Backup servers & data replication – Ensure no data loss due to hardware failure.
- Real-time monitoring & alerts – Instant notifications for security threats or anomalies**.**

4. Compliance & Regulatory Standards:

- PCI-DSS Compliance – Secure handling of credit card transactions.
- GDPR & CCPA Compliance – Protect user data and privacy.
- SOC 2 Type II Certification – Ensure best security practices for financial transactions.

5. Incident Response & Logging:

- Audit logs for all transactions – Maintain detailed records for security audits.
- Automated threat detection – AI-driven anomaly detection to prevent data breaches.
- Emergency shutdown protocol – Ability to pause transactions in case of system compromise.

5.3 Security Requirements

The system must implement robust security measures to protect user data, prevent unauthorized access, and ensure secure payment processing**.**

1. User Authentication & Access Control:

- Multi-Factor Authentication (MFA) – Users must verify identity using password + OTP/email verification.
- Role-Based Access Control (RBAC) – Users, admins, and support staff should have different levels of access.
- OAuth 2.0 / OpenID Connect (OIDC) – Secure API authentication using tokens.
- Account Lockout Policy – Lock accounts after 5 failed login attempts to prevent brute-force attacks.

2. Secure Data Storage & Transmission:

- End-to-End Encryption (E2EE) – All sensitive data must be encrypted using AES-256.
- TLS 1.3 with HSTS (HTTP Strict Transport Security) – Prevents MITM (Man-in-the-Middle) attacks.
- Tokenization for Card Data – Replace actual card details with secure tokens to prevent exposure.
- Database Encryption – Store user credentials and financial data in an encrypted format.

3. Fraud Prevention & Anomaly Detection:

- Real-Time Fraud Detection – Use AI/ML to flag suspicious transactions.
- Geolocation & Device Fingerprinting – Identify and block unusual login attempts.
- Rate Limiting & CAPTCHA – Prevent bot-driven attacks on login and payment forms.
- IP Whitelisting for Admin Access – Only allow admin logins from trusted networks.

4. Secure API Communication:

- RESTful APIs secured with OAuth 2.0 / JWT for authentication.
- CORS (Cross-Origin Resource Sharing) Policies – Restrict API access to trusted domains.
- API Rate Limiting – Prevent API abuse by setting request limits.

5. Compliance with Security Standards:

- PCI-DSS Compliance – Ensures safe handling of credit card transactions.
- GDPR & CCPA Compliance – Protects user privacy and data rights.
- SOC 2 Type II Certification – Ensures secure financial operations.

6. Incident Response & Monitoring:

- 24/7 Security Monitoring – Track system logs for security anomalies.
- Audit Logs for All Transactions – Maintain a tamper-proof transaction history.
- Automated Security Alerts – Notify administrators of suspicious activities.
- Penetration Testing & Ethical Hacking – Conduct regular security tests to identify vulnerabilities.

Appendix A: Glossary

1.ML (Machine Learning): A type of artificial intelligence that enables the system to learn from transaction data and improve scam detection over time.

2.Phishing: A fraudulent attempt to obtain sensitive information by disguising as a trustworthy entity in digital communication.

3.PCI DSS (Payment Card Industry Data Security Standard): A set of security standards designed to ensure that all companies that process, store, or transmit credit card information maintain a secure environment.

4.False Positive: A legitimate transaction that is incorrectly flagged as fraudulent.

5.Two-Factor Authentication (2FA): An additional layer of security that requires not only a password and username but also something the user has on them (like a smartphone).

Appendix B: Analysis Models

- **Use case diagrams.**
- **Sequence diagrams**.