User Fraud Check Module – Technical Design

1. Purpose

Design a lightweight but extensible **User Fraud Check module** that evaluates whether a user is potentially fraudulent. This module can be consumed as an API using a POST method.

2. Scope & Assumptions (IEEE 830 – What)

In scope - Evaluate a user using a set of fraud rules - Return an explainable fraud result - Mock external fraud services

Out of scope - Persistence - Machine learning–based fraud detection – External services.

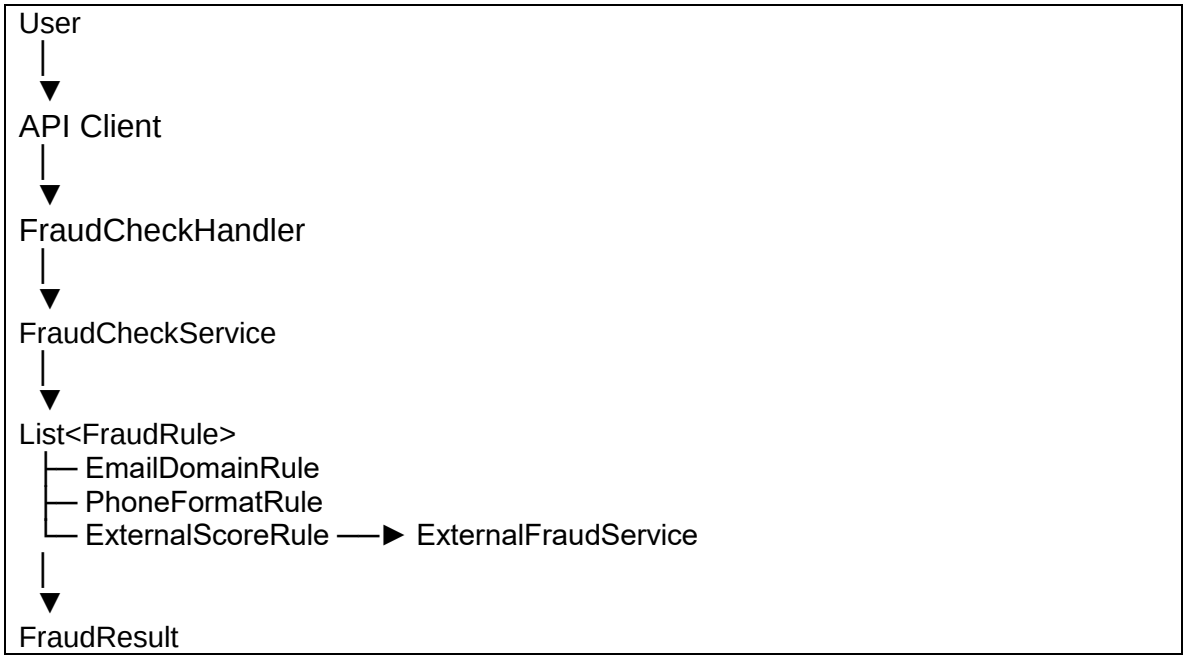
Assumptions - Checks are synchronous - Fraud detection is rule-based

3. High-Level Architecture (ISO/IEC/IEEE 42010)

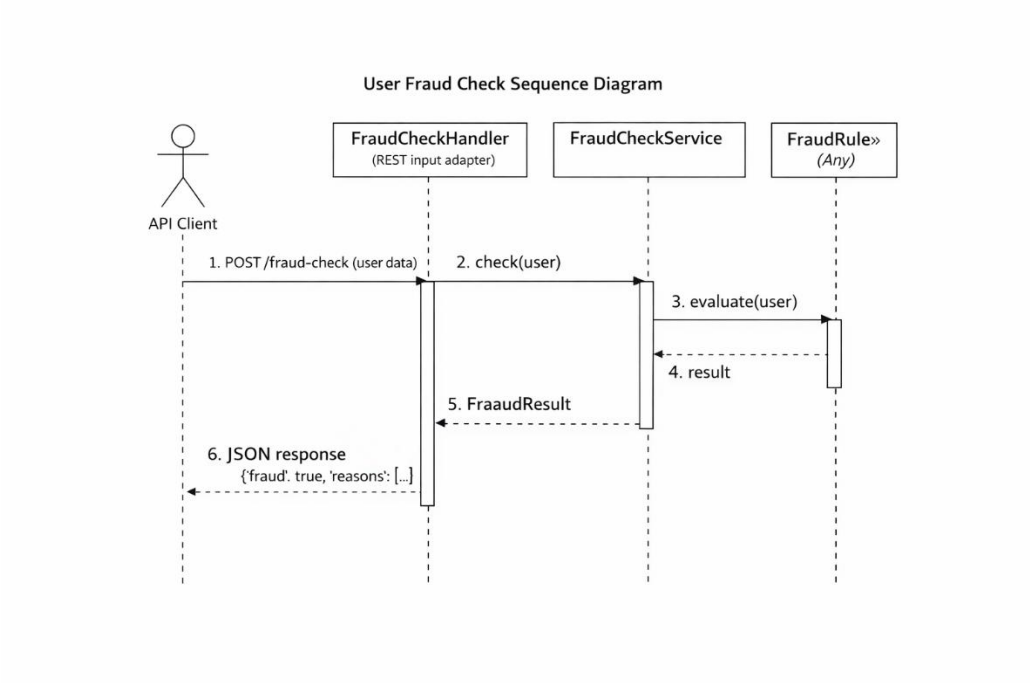
Stakeholders - Developers - Reviewers / interview panel

Architectural style - Rule-based component architecture

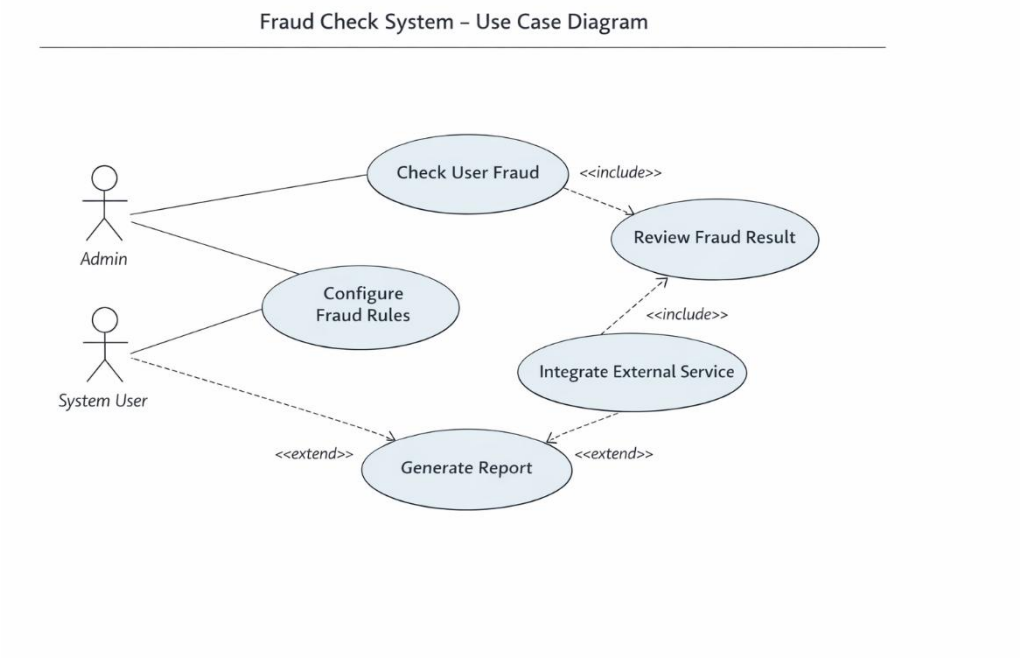
Component view



Sequence diagram



User Case diagram



4. Internal Design (IEEE 1016 – How)

Domain Model

- **User**: id, email, phone
- **FraudResult**: isFraud, reasons

Core Components

- **FraudRule (interface)**
  Encapsulates a single fraud check and returns a rule result.

- **FraudCheckService**
  Orchestrates rule execution and aggregates results. All rules are evaluated

Example Rules

- **EmailDomainRule**: detects disposable or suspicious email domains
- **PhoneFormatRule**: validates phone number format

5. Error Handling & Testing

Error handling - Invalid or missing user fields handled defensively - External service failures do not automatically mark fraud

Testing strategy - Unit tests per rule - Aggregation tests for FraudCheckService -

6. REST Exposure

The fraud check module is exposed via a REST API implemented using Java standard libraries only. The REST layer acts as a thin adapter and does not contain business logic.

Endpoint:
POST /fraud-check

Request:
{ "id": "1", "email": "test@mailinator.com", "phone": "+521234567890" }

Response:
{ "fraud": true, "reasons": ["Blacklisted email domain"] }

7. API Contract

The REST API is documented using OpenAPI 3.0 (Swagger), providing a clear contract between clients and the service. The specification is aligned 1:1 with the implemented endpoint.

8. Summary

This project implements a lightweight user fraud detection service in pure Java. It evaluates users using a rule-based engine and exposes the functionality via a REST API. Core fraud logic is framework-agnostic and fully unit-tested.