**📘 Payment Oversight Module – Thunglish Explanation**

**Backend Framework:** FastAPI

**1. System Architecture**

**1.1 Overview**

**English:**  
This module ensures secure handling of financial transactions, detects suspicious transactions, manages disputes, and keeps an audit trail.

**Thunglish Explanation:**

* இந்த module **financial transactions**-ஐ safe-a handle பண்ணும்.
* **Flagged transactions** (suspicious/strange transactions) detect பண்ணும்.
* Payment **disputes** resolve பண்ண workflow-ல் handle பண்ணும்.
* எல்லா actions-um **audit trail**-ல் log ஆகும் for transparency & compliance.

**Core Principles:**

* Security-first design → **முதலில் security**
* Regulatory compliance (AML/KYC, GDPR) → **laws & rules பூர்த்தி பண்ணு**
* Horizontal scalability → **பல servers-ல் scale பண்ணலாம்**
* End-to-end logging & monitoring → **actions track பண்ணலாம்**

**1.2 High-Level Architecture**

**English:**  
Frontend (dashboard) → Backend (FastAPI APIs) → Database → Notification + Audit Logging + Monitoring

**Thunglish:**

* **Frontend:** Angular/React dashboard admin-ku
* **Backend:** FastAPI REST APIs
* **Database:** PostgreSQL, Redis (cache frequent flagged transactions)
* **Notification Service:** Email, SMS, push alerts
* **Audit Logging Service:** Immutable logs (change pannanum illa)
* **Monitoring:** Prometheus & Grafana for metrics

**Flow:**

* Admin dashboard-ல transactions பார்க்கும் → API call → DB fetch → return to admin
* Dispute resolve-ஐ POST பண்ணும்போது → DB update → user-ku notify → audit log save

**2. Data Models (Database Tables)**

**2.1 Flagged Transactions Table**

| **Field** | **Type** | **Explanation** |
| --- | --- | --- |
| id | UUID | Unique identifier |
| transaction\_id | UUID | Which transaction flagged |
| flagged\_reason | TEXT | Reason (fraud, AML, anomaly) |
| flagged\_at | TIMESTAMP | When flagged |
| status | ENUM | pending / reviewed / cleared |

**Thunglish:**

* இது suspicious transactions-ஐ track பண்ண table
* **status** update பண்ணலாம் after review

**2.2 Payment Disputes Table**

| **Field** | **Type** | **Explanation** |
| --- | --- | --- |
| id | UUID | Unique dispute id |
| transaction\_id | UUID | Related transaction |
| user\_id | UUID | Who raised dispute |
| dispute\_reason | TEXT | Details of dispute |
| status | ENUM | open / under\_review / resolved / escalated |
| created\_at | TIMESTAMP | When created |
| resolved\_at | TIMESTAMP | When resolved |

**Thunglish:**

* User complaint handle பண்ண table
* **status** change according to workflow

**2.3 Audit Logs Table**

| **Field** | **Type** | **Explanation** |
| --- | --- | --- |
| id | UUID | Unique log id |
| admin\_id | UUID | Admin action perform பண்ணவர் |
| action\_type | ENUM | flagged\_view / dispute\_resolved / escalation / notification\_sent |
| target\_id | UUID | Which transaction or dispute affected |
| timestamp | TIMESTAMP | When action happened |
| metadata | JSONB | Extra info |

**Thunglish:**

* All admin actions **track** பண்ண table
* Cannot change old logs → **immutable**

**3. API Endpoints (FastAPI Style)**

**3.1 GET /flagged**

* **Purpose:** Get all pending flagged transactions
* **Thunglish:** Admin dashboard-ல pending flagged transactions list fetch பண்ண API
* **Errors:** 401 → invalid token, 404 → no transactions, 500 → server error

**3.2 POST /disputes/{id}/resolve**

* **Purpose:** Resolve a payment dispute
* **Thunglish:** Admin dispute resolve பண்ணும் API
* **Request Body Example:** { "resolution": "Transaction refunded", "resolved\_by": "admin-456" }
* **Response:** dispute status updated, resolved\_at set
* **Errors:** 400 → wrong data, 404 → dispute not found, 409 → already resolved, 500 → server error

**4. Dispute Resolution Workflow**

**Thunglish Explanation:**

1. Admin view flagged transactions → API fetch data → show in dashboard
2. Admin resolve dispute → API update DB → Notify user → Log in audit\_logs

**5. Security & Compliance**

* Encryption: AES-256, TLS 1.3
* JWT-based admin authentication (RBAC)
* GDPR / AML / KYC compliance
* PII Masking → sensitive user info hide in logs
* Immutable audit logs

**Thunglish:**

* Data safe-a store பண்ணும்
* Unauthorized access block பண்ணும்

**6. Error Handling**

* JSON structured error responses
* Retry failed notifications
* Prometheus alert repeated errors

**Thunglish:**

* Error details JSON-ல வந்து understandable-a இருக்க வேண்டும்
* Failed task retry பண்ணலாம்

**7. Performance Optimization**

* Indexes on transaction\_id & status
* DB connection pooling
* Redis caching
* Async queues for notifications (Celery / RQ)

**Thunglish:**

* Fast & efficient API calls
* Heavy load handle பண்ண ready

**8. Testing Strategy**

* Unit Tests → individual functions
* Integration Tests → end-to-end workflow
* Negative Tests → wrong IDs, unauthorized admin
* Performance Tests → load & stress test

**Thunglish:**

* எல்லா scenarios test பண்ணி bugs prevent பண்ணலாம்

**9. Deployment & Scaling**

* Docker + Kubernetes
* CI/CD → GitHub Actions + Helm
* Monitoring alerts
* Horizontal scaling for FastAPI services
* Future: ML fraud detection, chatbot disputes, blockchain audit

**Thunglish:**

* Production ready
* Scale பண்ணலாம் according to traffic

**10. Troubleshooting & Best Practices**

* Check audit logs if dispute mismatch
* Flagged reason human-readable இருக்க வேண்டும்
* Escalation workflow for old disputes
* Rate limiting prevent brute force

**Thunglish:**

* Admin mistakes, system issues easy-a debug பண்ணலாம்

✅ **Summary (Thunglish):**  
FastAPI backend **flagged transactions, dispute resolution, notifications, audit logs** handle efficiently, **async operations & caching support**, ready for scaling.