

«abstract»
User
userId: String # email: String # passwordHash: String # createdAt: DateTime # lastLogin: DateTime
+ login(email, pass): Token + logout(): void + updateProfile(data): Bool # validateCreds(pass): Bool

RegisteredUser
- disputeHistory: List<Dispute> - trustLevel: Float - phoneNumber: String
+ initiateDispute(order): Dispute + uploadEvidence(img, meta): Evid + viewStatus(id): Status + getNotifications(): List<Notif>

SupportAgent
- assignedDisputes: List<Dispute> - resolvedCount: Integer - department: String
+ reviewFraudAlert(id): void + requestEvidence(id): void + makeDecision(id, dec): Bool + viewPendingCases(): List<Dispute>

SystemAdmin
- permissions: List<String> - adminLevel: Integer
+ updateThresholds(w): Bool + viewAnalytics(): Report + configureSystem(set): Bool + genReport(): Report

RefundDispute
- disputeId: String - orderId: String - status: Status - createdAt: DateTime - trustScore: Float - assignedAgent: Agent
+ updateStatus(stat): void + getEvidence(): List<Evid> + calculateAge(): Duration + assignToAgent(agent): void

Evidence
- evidencId: String - imageData: Bytes - imageHash: String - metaScore: Float - elaScore: Float - aiScore: Float
+ generateHash(): String + isLiveCapture(): Bool + getScores(): ScoreTriple

ForensicEngine
- config: AnalysisConfig
+ analyze(evid): Result + runFullScan(img): Tuple - orchestrate(evid): void

MetadataAnalyzer
- suspiciousSoft: List<String> - requiredFields: List<String>
+ extractEXIF(img): Exif + detectSignatures(exif): Bool + validateSource(exif): Float

ELAProcessor
- quality: Int = 90 - ampFactor: Int = 50 - varianceThresh: Float
+ performELA(img): ELAResult + calcPixelDiff(a, b): Matrix + computeVariance(amp): Float

AISemanticChecker
- endpoint: String - apiKey: String - timeout: Integer
+ analyzeConsistency(img, txt): Float + sendToGemini(load): AIResp - buildPrompt(img, claim): String

TrustScoreCalculator
- weights: ScoreWeights - autoRefundThresh: Float - fraudAlertThresh: Float
+ calcTrustScore(m, e, a): Float + setWeights(w1, w2, w3): void + getAction(score): Type

DecisionRouter
- api: RefundAPI - notify: NotificationService
+ routeDecision(dis, score): void + processAutoRefund(d): Bool + createFraudAlert(d): Alert

«interface»
RefundAPI
- endpoint: String - token: String
+ initiateRefund(id, amt): Resp + checkStatus(id): Status + cancelRefund(id): Bool

«interface»
NotificationService
- emailProvider: String - smsProvider: String
+ sendEmail(to, sub, body): Bool + sendSMS(num, msg): Bool + sendPush(id, msg): Bool