



YOUR JOURNEY, SIMPLIFIED

SECURITY AUDIT DOCUMENTATION

Web Application Security Assessment

Comprehensive technical documentation package for INSA cybersecurity compliance audit

PREPARED FOR

Information Network Security Administration (INSA)

Cyber Security Audit Division

APPLICATION

i-Ticket Online Bus Ticketing Platform

VERSION

v2.14.0

PRODUCTION URL

<https://i-ticket.et>

DATE

February 2026

DOCUMENTS INCLUDED

17 Technical Documents

CLASSIFICATION

Confidential



Confidential Document — This document contains proprietary technical information about the i-Ticket platform's security architecture, API endpoints, authentication mechanisms, and system configuration. Distribution is restricted to authorized INSA audit personnel only.





DOCUMENT

4.2.1a

Data Flow Diagram

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Data Flow Diagrams (DFD)

Document: 4.2.1a - Data Flow Diagram Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Context-Level DFD (Level 0)

The Level 0 DFD shows the i-Ticket system as a single process interacting with all external entities.

External Entities

ID	Entity	Type	Description
E1	Registered Customer	Human	End users who register with phone + password and book bus tickets
E2	Guest Customer	Human	Users who book without registration (phone payment = verification)
E3	Company Admin	Human	Bus company administrators managing trips, staff, vehicles, fleet
E4	Super Admin	Human	i-Ticket platform administrators with full system access
E5	Driver	Human	Bus drivers providing GPS location and managing trip departures
E6	Conductor	Human	Bus conductors verifying tickets and managing boarding
E7	Cashier / Manual Ticketer	Human	Station staff selling tickets manually (cash payments)
E8	Mechanic	Human	Fleet maintenance personnel performing inspections and repairs
E9	Finance Staff	Human	Company finance personnel viewing reports and costs
E10	Sales Person	Human	Affiliate marketers recruiting customers via referral codes
E11	Platform Staff	Human	i-Ticket employees (support, accountant, DevOps)
S1	TeleBirr Payment Gateway	External System	Mobile money payment processing (Ethio Telecom)
S2	SMS Gateway (Negarit/GeezSMS)	External System	SMS delivery for ticket confirmations, reminders, booking
S3	Telegram Bot API	External System	Telegram messaging platform for bot-based booking
S4	ClickUp API	External System	Project management for support tickets and alerts
S6	OpenStreetMap Tiles	External System	Map tile imagery for tracking maps
S7	OSRM Routing API	External System	Road route geometry for map overlays
S8	Nominatim Geocoding	External System	Reverse geocoding (coordinates to address names)
S9	Cloudflare CDN/WAF	External System	SSL termination, DDoS protection, HSTS, caching
S12	OsmAnd GPS App	External System	Background GPS tracking from driver phones
S14	Cron Scheduler	External System	Scheduled job triggers (cleanup, reminders, AI scoring)

Level 0 Diagram



Data Flows Summary (Level 0)

Flow	From	To	Data Description
F1	E1/E2	P0	Trip search queries (origin, destination, date), booking requests (passengers, seats), payment initiation
F2	P0	E1/E2	Available trips, seat maps, booking confirmations, tickets with QR codes, bus GPS location

F3	E3	P0	Trip CRUD, staff management, vehicle management, fleet analytics queries, report requests
F4	P0	E3	Trip data, booking data, manifests, analytics dashboards, Excel reports, fleet tracking map
F5	E4	P0	Company management, trip oversight, system analytics, platform staff management
F6	P0	E4	Dashboard stats, audit logs, tax reports, revenue analytics, all company data
F7	E5	P0	GPS coordinates (lat, lon, altitude, accuracy, heading, speed), trip status updates
F8	P0	E5	Trip assignment, ETA calculations, tracking status
F9	E6	P0	Ticket QR scans (shortCode), boarding status updates
F10	P0	E6	Ticket verification results, passenger details, trip manifests
F11	E7	P0	Manual ticket sales (cash), replacement ticket sales for no-shows
F12	P0	E7	Booking confirmations, tickets with QR codes and shortCodes
F13	P0	S1	Payment initiation (phone, amount, merchantCode, HMAC signature)
F14	S1	P0	Payment callback (transactionId, status, amount, signature)
F15	P0	S2	Outbound SMS (ticket confirmations, reminders, payment timeout notifications)
F16	S2	P0	Inbound SMS webhook (booking commands from customers)
F17	S3	P0	Telegram webhook updates (user messages, button callbacks, location shares)
F18	P0	S3	Bot responses (text, inline keyboards, location messages, ticket images)
F19	P0	S4	Support ticket creation, low-slot alerts, audit log tasks
F20	S12	P0	Background GPS from OsmAnd app (token-authenticated GET requests)
F21	S14	P0	Cron trigger (Bearer token authenticated GET requests)
F22	P0	S10	All database read/write operations via Prisma ORM

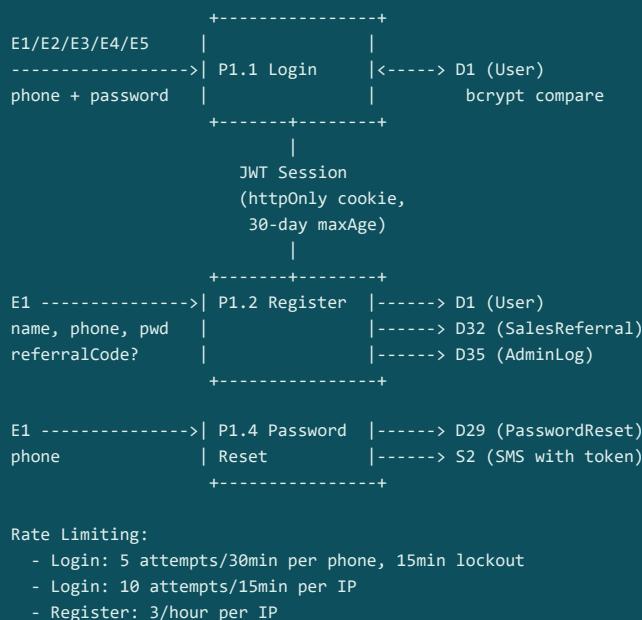
2. Level 1 DFD - Core Business Processes

The Level 1 DFD decomposes the i-Ticket system into its major functional processes.

Process Decomposition

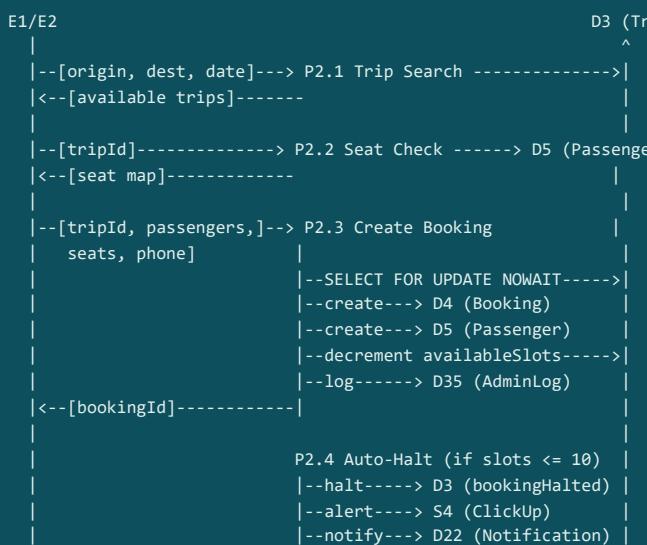


P1: Authentication & Session Management

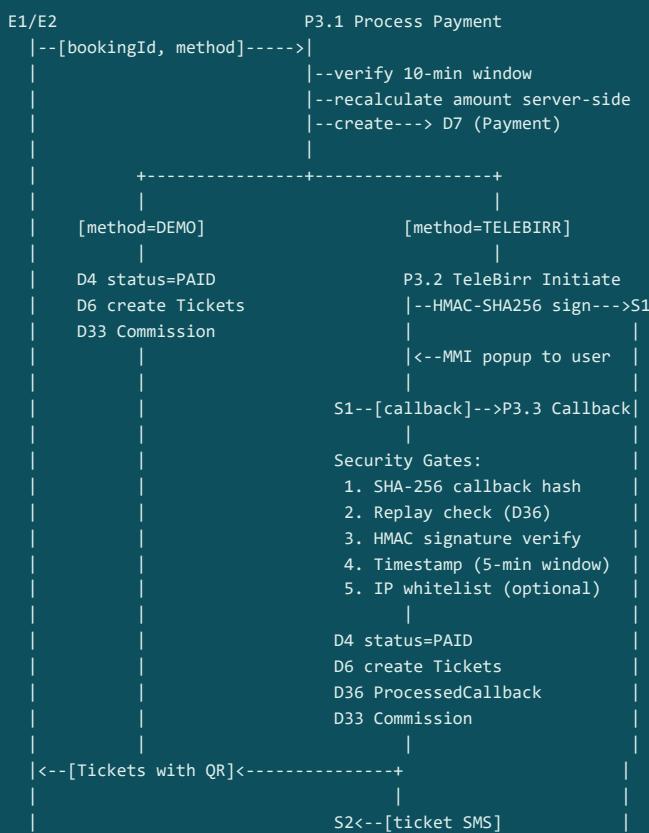




P2: Trip Search & Booking

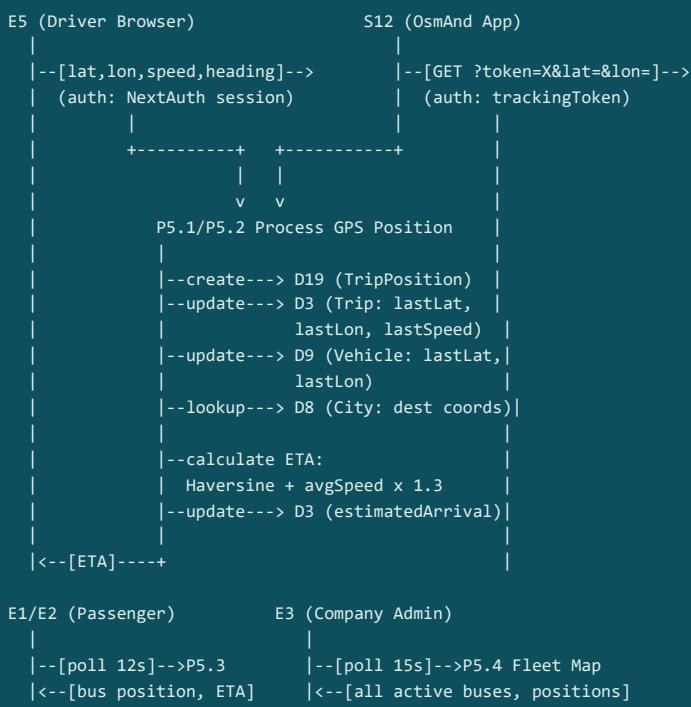


P3: Payment Processing

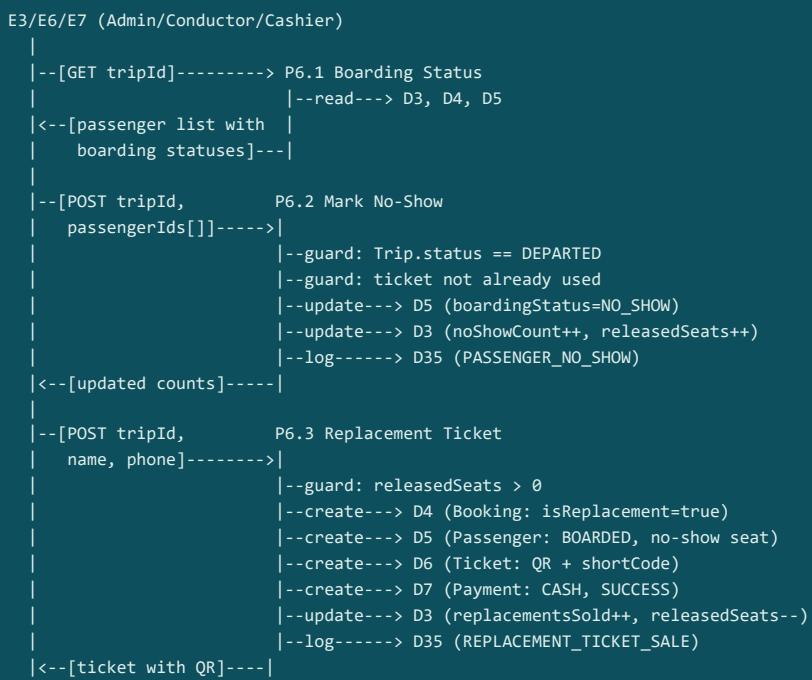




P5: GPS Tracking



P6: No-Show & Replacement Ticket Flow



3. Level 2 DFD - Detailed Sub-Processes

P3.3 TeleBirr Callback Processing (Detailed)

This is the most security-critical data flow in the system.



```
S1 (TeleBirr)
|
| --[POST /api/payments/telebirr/callback]
|   Body: { transactionId, outTradeNo, totalAmount,
|           tradeStatus, signature, timestamp, nonce }
|
| v
+-----+
| P3.3.1 Compute Callback Hash
| SHA-256(JSON.stringify(sortedPayload))
+-----+
|   |
|   v
+-----+
| P3.3.2 Replay Detection
| Query D36 (ProcessedCallback) by transactionId
| IF exists: return 200 (idempotent) without processing
+-----+
|   |
|   v
+-----+
| P3.3.3 Signature Verification
| HMAC-SHA256(payload, TELEBIRR_APP_KEY)
| crypto.timingSafeEqual(computedSig, receivedSig)
| IF mismatch: log + return 403
+-----+
|   |
|   v
+-----+
| P3.3.4 Timestamp Validation
| |now - callbackTimestamp| < 5 minutes
| IF expired: log warning (still process - clock skew)
+-----+
|   |
|   v
+-----+
| P3.3.5 IP Whitelist Check (Optional)
| IF TELEBIRR_WEBHOOK_IPS env set:
| Verify request IP in allowed list
| IF blocked: return 403
+-----+
|   |
|   v
+-----+
| P3.3.6 Transaction Processing (10s timeout)
|
| BEGIN TRANSACTION
|   SELECT Booking FOR UPDATE NOWAIT (row lock)
|   IF tradeStatus == SUCCESS:
|     UPDATE Payment SET status=SUCCESS
|     UPDATE Booking SET status=PAID
|     CREATE Ticket[] (QR code + shortCode per passenger)
|     CREATE SalesCommission (if referred user)
|     INSERT ProcessedCallback (idempotency record)
|   ELSE:
|     UPDATE Payment SET status=FAILED
|     UPDATE Booking SET status=CANCELLED
|     UPDATE Trip SET availableSlots += releasedSeats
|     INSERT ProcessedCallback
| COMMIT
|
| POST-COMMIT:
|   Send ticket SMS via S2
|   Log to D35 (AdminLog)
+-----+
```



P13 Cron Jobs (Detailed Lifecycle)

```
S14 (Cron Scheduler)
|
| --[GET /api/cron/cleanup, Bearer: CRON_SECRET]
|
| v
+-----+
| P13.1 Payment/Booking Cleanup (every 15 min)
|
| 1. Find PENDING payments older than 5 min
|    -> D7: status=TIMEOUT
|    -> D4: status=CANCELLED
|    -> D3: availableSlots += released seats
|    -> S2: Send timeout SMS to customer
|
| 2. Find PENDING bookings older than 10 min
|    -> D4: status=CANCELLED
|    -> D7: cancel associated payment
|    -> D3: availableSlots += released seats
+-----+
+
+-----+
| P13.2 Trip Status Lifecycle (every 15 min)
|
| SCHEDULED + 30min late -----> DELAYED
| SCHEDULED/BOARDING/DELAYED      DEPARTED
|   + past departure time -----> (bookingHalted=true)
|     D1: staff -> ON_TRIP
| DEPARTED + past ETA + 2hr -----> COMPLETED
|   (trackingActive=false)
|   D1: staff -> AVAILABLE
| SCHEDULED/BOARDING + 24hr old   CANCELLED
|   + zero bookings -----> (cleanup)
+-----+
+
+-----+
| P13.4 GPS Data Retention (every 15 min)
|
| DELETE FROM TripPosition
| WHERE trip.status IN (COMPLETED, CANCELLED)
| AND recordedAt < NOW() - 7 days
+-----+
+
+-----+
| P13.6 Predictive Maintenance (daily)
|
| FOR EACH Vehicle:
|   Calculate risk score (0-100) based on:
|     - Odometer since last service
|     - Time since last service
|     - Defect history from inspections
|     - Fuel efficiency trend
|     - Vehicle age and total mileage
|   -> D9: update riskScore, predictedFailureDate
|   -> D17: create VehicleRiskHistory snapshot
+-----+
```

4. Data Store Summary

4.1 Core Business Data

Store ID	Name	Sensitivity	Encrypted Fields
D1	User	HIGH	password (bcrypt hash)
D2	Company	HIGH	bankAccount, tinNumber

D3	Trip	MEDIUM	-
D4	Booking	HIGH	financial amounts
D5	Passenger	HIGH	nationalId, phone
D6	Ticket	HIGH	shortCode (auth token)
D7	Payment	HIGH	transactionId, amounts
D8	City	LOW	-

4.2 Fleet Management Data

Store ID	Name	Sensitivity
D9	Vehicle	MEDIUM
D10	TripLog	MEDIUM
D11	MaintenanceSchedule	LOW
D12	WorkOrder	MEDIUM
D13	WorkOrderPart	LOW
D14	VehicleInspection	MEDIUM
D15	FuelEntry	MEDIUM
D16	OdometerLog	LOW
D17	VehicleRiskHistory	LOW
D18	VehicleDowntime	LOW

4.3 GPS & Communication Data

Store ID	Name	Sensitivity
D19	TripPosition	MEDIUM (location tracking)
D20	TripMessage	MEDIUM
D22	Notification	LOW
D23	CompanyMessage	MEDIUM
D26	SupportTicket	HIGH (PII)

4.4 Session & Security Data

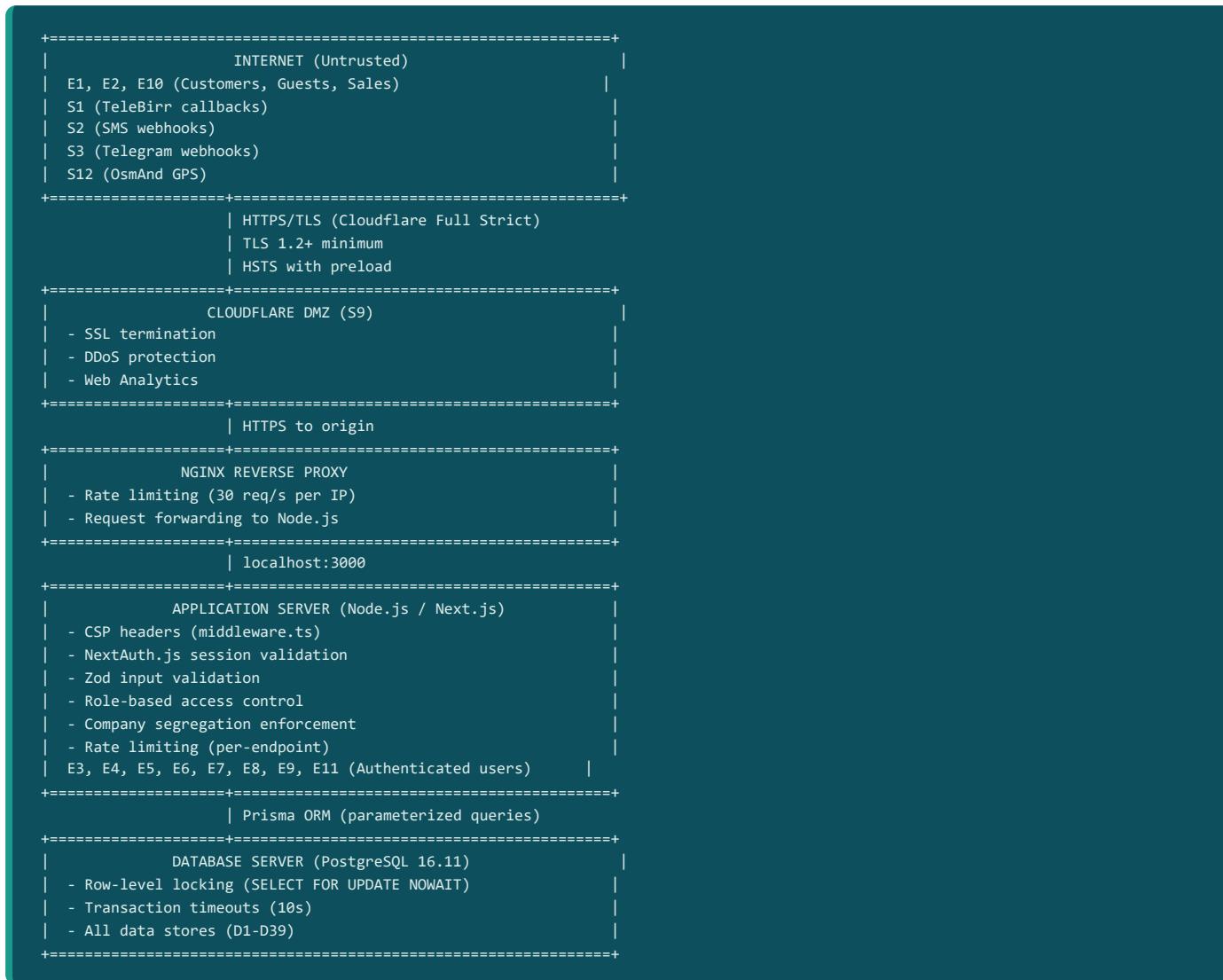
Store ID	Name	Sensitivity
D27	SmsSession	MEDIUM
D28	TelegramSession	MEDIUM
D29	PasswordReset	HIGH (auth tokens)
D35	AdminLog	HIGH (audit trail)
D36	ProcessedCallback	HIGH (payment security)

4.5 Sales & Financial Data

Store ID	Name	Sensitivity
D30	SalesPerson	HIGH (PII, financial)
D33	SalesCommission	HIGH (financial)

D34	SalesPayout	HIGH (financial)
-----	-------------	------------------

5. Trust Boundaries



6. Entry Points

All external inputs to the system that INSA should test:

#	Entry Point	Protocol	Auth Required	Rate Limited
1	/api/auth/[...nextauth] (login/register)	HTTPS POST	No	Yes (5/30min phone, 10/15min IP)
2	/api/trips (search)	HTTPS GET	No	Nginx 30r/s
3	/api/bookings (create)	HTTPS POST	Optional (guest allowed)	Yes (10/min/IP)
4	/api/payments (initiate)	HTTPS POST	Yes (session)	Yes (3/hour/booking)
5	/api/payments/telebirr/callback	HTTPS POST	HMAC signature	Yes + replay guard
6	/api/tickets/verify	HTTPS PATCH	Yes (staff)	Nginx 30r/s

7	/api/track/[code] (public tracking)	HTTPS GET	No	Nginx 30r/s
8	/api/tracking/update (driver GPS)	HTTPS POST	Yes (session)	Yes (12/min/user)
9	/api/tracking/osmand (background GPS)	HTTPS GET	Token	Yes (12/min/token)
10	/api/tracking/[tripId] (passenger tracking)	HTTPS GET	No	Nginx 30r/s
11	/api/tracking/fleet (fleet map)	HTTPS GET	Yes (company admin)	Nginx 30r/s
12	/api/telegram/webhook	HTTPS POST	Telegram-signed	Nginx 30r/s
13	/api/sms/incoming	HTTPS POST	HMAC signature	Yes (10/min/phone)
14	/api/company/* (all company routes)	HTTPS Various	Yes (company admin)	Nginx 30r/s
15	/api/admin/* (all admin routes)	HTTPS Various	Yes (super admin)	Nginx 30r/s
16	/api/cron/* (scheduled jobs)	HTTPS GET	Bearer token	Nginx 30r/s
17	/api/support/tickets (create)	HTTPS POST	No	Nginx 30r/s
18	/api/cities/coordinates (geocoding)	HTTPS GET	No	Nginx 30r/s

End of Document 4.2.1a - Data Flow Diagram





DOCUMENT

4.2.1b

System Architecture Diagram

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION



i-Ticket Platform - System Architecture Diagram

Document: 4.2.1b - System Architecture Diagram Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform
URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Deployment Architecture

i-Ticket uses a **cloud-hosted** deployment on AWS with Cloudflare as the CDN/WAF layer.



1.1 Infrastructure Overview

```
+=====+  
| INTERNET  
| Customers, Guests, Drivers, Staff, Telegram, TeleBirr, OsmAnd |  
+=====+  
|  
| DNS: i-ticket.et  
| (Cloudflare nameservers)  
|  
+=====+  
| CLOUDFLARE CDN / WAF  
|  
| Plan: Free  
| SSL Mode: Full (Strict)  
| TLS Minimum: 1.2  
| HSTS: Enabled (max-age=31536000, includeSubDomains, preload)  
| DDoS Protection: Automatic  
| Web Analytics: Enabled (beacon.min.js)  
| Email Obfuscation: DISABLED (breaks React hydration)  
| Caching: Standard (static assets)  
|  
| Nameservers:  
| - Cloudflare-assigned NS records  
| DNS Records:  
| - A record: i-ticket.et -> 54.147.33.168 (proxied)  
| - CNAME: www -> i-ticket.et (proxied)  
+=====+  
|  
| HTTPS (Cloudflare -> Origin)  
| (Cloudflare Origin Certificate)  
|  
+=====+  
| AWS EC2 INSTANCE (t2.micro)  
|  
| IP: 54.147.33.168  
| Region: us-east-1  
| OS: Ubuntu 22.04 LTS  
| CPU: 1 vCPU  
| RAM: 1 GB  
| Storage: 7.6 GB EBS (General Purpose SSD)  
| Security Group: SSH (22), HTTP (80), HTTPS (443)  
|  
+-----+  
| NGINX REVERSE PROXY  
|  
| Listen: Port 80 (HTTP)  
| Upstream: localhost:3000  
| Rate Limiting:  
| - General: 100 req/min per IP  
| - API: 60 req/min per IP  
| Max Body Size: 5 MB  
| Server Tokens: OFF (version hidden)  
| Headers Stripped: Server, X-Powered-By  
| Keepalive: 64 connections to upstream  
| Proxy Timeouts: 60s (connect, send, read)  
| Static Assets: /_next/static cached 60min  
+-----+  
|  
| localhost:3000  
|  
+-----+  
| PM2 PROCESS MANAGER  
|  
| App Name: i-ticket  
| Mode: Cluster (1 instance for t2.micro)  
| Auto-restart: Yes  
| Max Memory: 800 MB (restart threshold)  
| Kill Timeout: 5000ms (graceful shutdown)  
| Cron Restart: Daily at 03:00 UTC (memory leak prevention)  
| Max Restarts: 10  
| Min Uptime: 10s  
| Logs: /var/log/pm2/i-ticket-{error,out}.log  
|
```

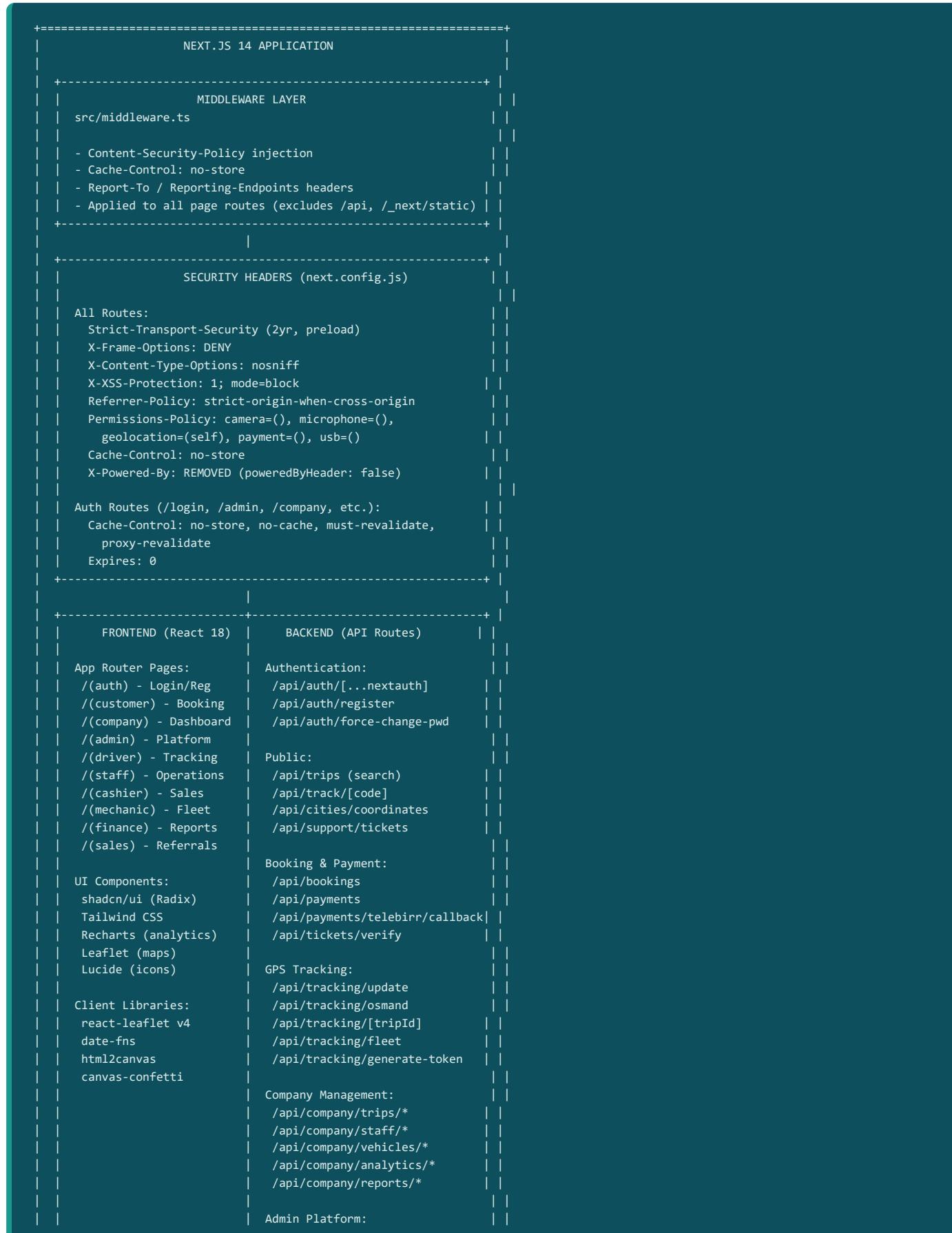
```
| Log Format: YYYY-MM-DD HH:mm:ss Z
+-----+-----+
|           |
|           |
+-----+-----+
|           NODE.JS 20.20.0 / NEXT.JS 14
|           |
|           |
| Runtime: Node.js 20.20.0 (LTS)
| Framework: Next.js 14 (App Router)
| Port: 3000
| Environment: NODE_ENV=production
| Working Dir: /var/www/i-ticket
+-----+-----+
|           |
|           |
|           |
|           Prisma ORM
|           (parameterized queries)
|           |
+-----+-----+
|           POSTGRESQL 16.11
|           |
|           |
| Host: localhost
| Port: 5432
| Connection: Unix socket (local)
| Auth: Password (via DATABASE_URL)
| Tables: 39 models
| Row-level Locking: SELECT FOR UPDATE NOWAIT
| Transaction Timeouts: 10 seconds
+-----+-----+
```

1.2 Infrastructure Specifications

Component	Specification
Cloud Provider	Amazon Web Services (AWS)
Instance Type	EC2 t2.micro (1 vCPU, 1 GB RAM)
Region	us-east-1 (N. Virginia)
Public IP	54.147.33.168
Operating System	Ubuntu 22.04 LTS
Storage	7.6 GB EBS (General Purpose SSD gp2)
Node.js	v20.20.0 (LTS)
Database	PostgreSQL 16.11 (local)
Process Manager	PM2 (cluster mode, 1 instance)
Reverse Proxy	Nginx
CDN / WAF	Cloudflare (Free plan)
SSL/TLS	Cloudflare Full (Strict) mode, TLS 1.2 minimum
Domain	i-ticket.et (registered via .et registry)
SSH Access	Key-based only (mela-shared-key.pem), port 22

2. Component Architecture

2.1 Application Layer Stack



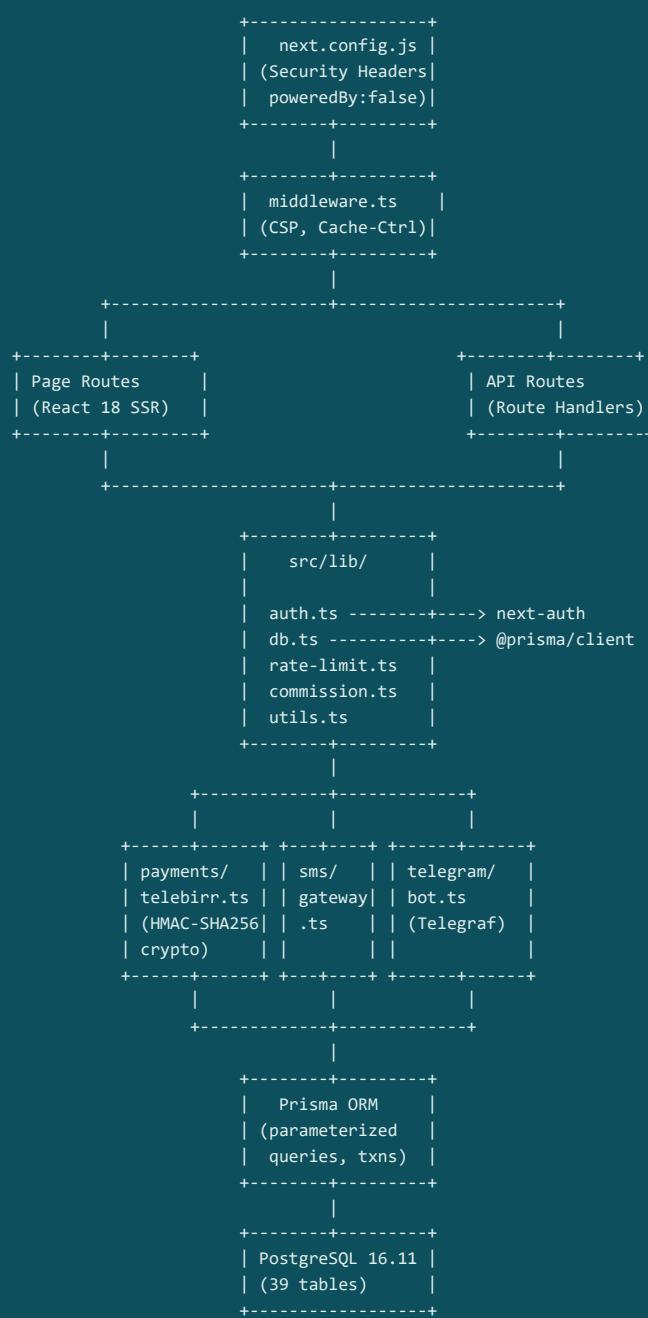
```
|   |   |   |   |   /api/admin/stats  
|   |   |   |   |   /api/admin/companies  
|   |   |   |   |   /api/admin/trips  
|   |   |   |   |   /api/admin/bookings  
|  
|   |   |   | Webhooks (Inbound):  
|   |   |   |   /api/telegram/webhook  
|   |   |   |   /api/sms/incoming  
|  
|   |   |   | Cron Jobs:  
|   |   |   |   /api/cron/cleanup  
|   |   |   |   /api/cron/trip-reminders  
|   |   |   |   /api/cron/predictive-maint  
|   |   |   |   /api/cron/telegram-cleanup  
|  
|   |   |   | Security:  
|   |   |   |   /api/csp-report  
+-----+-----+  
|  
+-----+-----+  
|           SHARED LIBRARIES (src/lib/)  
|  
| auth.ts      - NextAuth config, rate limiting  
| db.ts        - Prisma client singleton  
| rate-limit.ts - In-memory rate limiter  
| commission.ts - Commission & VAT calculation  
| utils.ts      - Ethiopia timezone, formatting  
| fuzzy-match.ts - Levenshtein fuzzy search  
|  
| payments/  
|   telebirr.ts    - TeleBirr API client, HMAC signing  
|   callback-hash.ts - SHA256 callback deduplication  
|  
| sms/  
|   gateway.ts     - SMS send/receive, webhook verification  
|  
| telegram/  
|   bot.ts        - Telegram bot (Telegraf framework)  
|   messages.ts    - Bilingual message templates  
|   keyboards.ts   - Inline keyboard builders  
|   middleware/auth.ts - Session management  
|   handlers/       - Command, payment, ticket, tracking  
|   scenes/         - Booking wizard state machine  
|  
| tracking/  
|   update-position.ts - Shared GPS processing logic  
|   audio-keep-alive.ts - Background GPS persistence  
|  
| ai/  
|   predictive-maintenance.ts - Vehicle risk scoring  
|  
| clickup/  
|   client.ts      - ClickUp API integration  
+-----+  
|  
+-----+-----+  
|           DATA ACCESS (Prisma ORM)  
|  
|   - Parameterized queries (SQL injection prevention)  
|   - Transaction support with configurable timeouts  
|   - Row-level locking (SELECT FOR UPDATE NOWAIT)  
|   - Connection pooling (Prisma default pool)  
|   - Schema migrations via `prisma db push`  
|  
+-----+-----+
```

2.2 Service-to-Service Communication

Source	Destination	Protocol	Auth Method	Direction	Data Format
Next.js App	PostgreSQL	TCP/Unix socket	DATABASE_URL credential	Bidirectional	SQL (via Prisma)

Next.js App	TeleBirr API	HTTPS	HMAC-SHA256 (APP_KEY)	Outbound	JSON
TeleBirr API	Next.js App	HTTPS	HMAC-SHA256 signature	Inbound (callback)	JSON
Next.js App	SMS Gateway	HTTPS	Bearer token (API_KEY)	Outbound	JSON
SMS Gateway	Next.js App	HTTPS	HMAC-SHA256 (WEBHOOK_SECRET)	Inbound (webhook)	JSON
Telegram API	Next.js App	HTTPS	Telegram-signed	Inbound (webhook)	JSON
Next.js App	Telegram API	HTTPS	Bot token	Outbound	JSON
Next.js App	ClickUp API	HTTPS	API token (Bearer)	Outbound	JSON
Next.js App	OSRM API	HTTPS	None (public)	Outbound	JSON
Next.js App	Nominatim API	HTTPS	None (public)	Outbound	JSON
Browser (client)	OSM Tile Server	HTTPS	None (public)	Outbound	PNG tiles
Cron Scheduler	Next.js App	HTTPS	Bearer token (CRON_SECRET)	Inbound	None (GET)
OsmAnd App	Next.js App	HTTPS	Trip tracking token (query param)	Inbound	URL params
Browser (client)	Next.js App	HTTPS	JWT cookie (NextAuth)	Bidirectional	JSON

2.3 Module Dependency Map



3. Security Layers

3.1 Defense-in-Depth Architecture

```
LAYER 1: EDGE SECURITY (Cloudflare)
+-----+
| Cloudflare CDN / WAF |
| +-----+ |
| | TLS 1.2+ Termination (Full Strict mode) |
| | HSTS: max-age=31536000; includeSubDomains; preload |
| | DDoS Protection (automatic) |
| | Bot Management (basic) |
| | Web Analytics Beacon |
| | Email Obfuscation: DISABLED (React hydration conflict) |
| | Origin Certificate: Cloudflare-issued |
| +-----+ |
+-----+
|
LAYER 2: NETWORK SECURITY (Nginx)
+-----+
| Nginx Reverse Proxy |
| +-----+ |
| | Rate Limiting: 100 req/min (general), 60 req/min (API) |
| | Server Header: STRIPPED (server_tokens off) |
| | X-Powered-By: STRIPPED |
| | Client Max Body Size: 5 MB |
| | Proxy Headers: X-Real-IP, X-Forwarded-For, X-Forwarded-Proto |
| | Keepalive Connections: 64 to upstream |
| | Timeouts: 60s connect/send/read |
| +-----+ |
+-----+
|
LAYER 3: APPLICATION SECURITY (Next.js Middleware)
+-----+
| Next.js Middleware (src/middleware.ts) |
| +-----+ |
| | Content-Security-Policy: |
| | default-src 'self' |
| | script-src 'self' 'unsafe-inline' |
| | https://static.cloudflareinsights.com |
| | style-src 'self' 'unsafe-inline' |
| | https://fonts.googleapis.com |
| | img-src 'self' data: https://api.qrserver.com |
| | https://*.tile.openstreetmap.org https://unpkg.com |
| | font-src 'self' data: https://fonts.gstatic.com |
| | connect-src 'self' https://api.telebirr.com |
| | https://api.webbirr.com https://*.tile.openstreetmap.org |
| | https://nominatim.openstreetmap.org |
| | https://cloudflareinsights.com |
| | https://router.project-osrm.org |
| | media-src 'self' data: blob: |
| | frame-ancestors 'none' |
| | base-uri 'self' |
| | form-action 'self' |
| | object-src 'none' |
| | upgrade-insecure-requests |
| | report-uri /api/csp-report |
| | |
| | Cache-Control: no-store |
| | Pragma: no-cache |
| | CSP Violation Reporting: /api/csp-report (logged to PM2) |
| +-----+ |
+-----+
|
LAYER 4: APPLICATION SECURITY (next.config.js Headers)
+-----+
| Static Security Headers |
| +-----+ |
| | Strict-Transport-Security: max-age=63072000; includeSubDomains | |
| | | |
| | | preload (2 years) |
| | | |
| | | X-Frame-Options: DENY |
| | | |
| | | X-Content-Type-Options: nosniff |
| | |
```



```
| | X-XSS-Protection: 1; mode=block | |
| | Referrer-Policy: strict-origin-when-cross-origin | |
| | Permissions-Policy: camera=(), microphone=(), | |
| | geolocation=(self), payment=(), usb=(), bluetooth=(), | |
| | magnetometer=(), gyroscope=(), accelerometer=() | |
| | X-Powered-By: REMOVED | |
| +-----+ | |
+=====+ | |
| | LAYER 5: AUTHENTICATION & AUTHORIZATION | |
+=====+ | |
| | NextAuth.js (Credentials Provider) | |
| +-----+ | |
| | Strategy: JWT (stateless, 30-day maxAge) | |
| | Cookie: httpOnly, secure, sameSite=lax, path=/ | |
| | Password: bcryptjs (12 salt rounds) | |
| | Secret: NEXTAUTH_SECRET (min 32 chars, validated at startup) | |
| | Production Guard: rejects placeholder secrets | |
| | | |
| | Rate Limiting (In-Memory): | |
| | Login: 5 attempts/30min per phone (15min lockout) | |
| | Login: 10 attempts/15min per IP | |
| | Register: 3/hour per IP | |
| | Booking: 10/min per IP | |
| | Payment: 3/hour per booking | |
| | Password Reset: 3/hour per IP | |
| | Store: 100K max entries, emergency cleanup at 80% | |
| | | |
| | Role-Based Access Control (RBAC): | |
| | Roles: CUSTOMER, COMPANY_ADMIN, SUPER_ADMIN | |
| | Staff Roles: ADMIN, DRIVER, CONDUCTOR, MANUAL_TICKETER, | |
| | MECHANIC, FINANCE | |
| | Company Segregation: All company APIs filter by companyId | |
| +-----+ | |
+=====+ | |
| | LAYER 6: INPUT VALIDATION | |
+=====+ | |
| | Zod Schema Validation | |
| +-----+ | |
| | All API inputs validated with Zod schemas | |
| | Server-side amount recalculation (never trust client) | |
| | Phone format: /~09\d{8}~/ (Ethiopian format) | |
| | parseInt validation (rejects scientific notation) | |
| | .nullish() for searchParams (returns null, not undefined) | |
| +-----+ | |
+=====+ | |
| | LAYER 7: PAYMENT SECURITY | |
+=====+ | |
| | TeleBirr Integration Security | |
| +-----+ | |
| | Outbound: HMAC-SHA256 signed requests (APP_KEY) | |
| | Inbound Callback Verification: | |
| | 1. SHA-256 callback hash computation | |
| | 2. Replay detection (ProcessedCallback table) | |
| | 3. HMAC-SHA256 signature (crypto.timingSafeEqual) | |
| | 4. Timestamp validation (5-minute window) | |
| | 5. Optional IP whitelist (TELEBIRR_WEBHOOK_IPS) | |
| | Nonces: crypto.randomBytes(16) | |
| | Idempotency: unique transactionId + callbackHash indexes | |
| | Demo Mode: blocked in NODE_ENV=production | |
| +-----+ | |
+=====+ | |
| | LAYER 8: DATABASE SECURITY | |
+=====+ | |
| | PostgreSQL 16.11 + Prisma ORM | |
| +-----+ | |
| | Parameterized queries (Prisma ORM - SQL injection prevention) | |
| | Row-level locking: SELECT FOR UPDATE NOWAIT | |
| | Transaction timeouts: 10s (dual: Promise.race + Prisma native) | |
| | Transaction max wait: 5s | |
| | Passwords: bcrypt hashed (never plaintext) | |
```



```
| | Reset tokens: bcrypt hashed before storage      | |
| | Connection: localhost only (no external access) | |
| | Logging: errors only in production           | |
| +-----+                                         | |
+=====+
|
LAYER 9: AUDIT & MONITORING
+=====+
| Audit Trail & Logging                         | | |
| +-----+                                         | |
| | AdminLog: all sensitive operations logged with userId, action, | |
| | details, tripId, companyId, timestamp          | |
| | ProcessedCallback: payment callback audit trail | |
| | CSP Violation Reports: logged to PM2 via /api/csp-report | |
| | PM2 Logs: /var/log/pm2/i-ticket-{error,out}.log       | |
| | Nginx Logs: access.log, error.log              | |
| | Cloudflare: Web Analytics, security events     | |
| +-----+                                         | |
+=====+
```

3.2 Security Controls Matrix

Control Category	Implementation	Standard
TLS/SSL	Cloudflare Full Strict, TLS 1.2 minimum, HSTS preload	OWASP A02, NIST SP 800-52
WAF/DDoS	Cloudflare automatic DDoS protection	OWASP A06
Rate Limiting	3-tier: Cloudflare -> Nginx -> Application	OWASP A04
CSP	Strict policy with report-uri, frame-ancestors 'none'	OWASP A03
Authentication	bcrypt passwords, JWT sessions, rate-limited login	OWASP A07
Authorization	RBAC with company segregation, staff sub-roles	OWASP A01
Input Validation	Zod schemas on all API inputs	OWASP A03
SQL Injection	Prisma ORM (parameterized queries)	OWASP A03
XSS	React auto-escaping + CSP	OWASP A03
CSRF	SameSite=lax cookies + form-action CSP	OWASP A01
Clickjacking	X-Frame-Options: DENY + frame-ancestors 'none'	OWASP A05
MIME Sniffing	X-Content-Type-Options: nosniff	OWASP A05
Information Disclosure	X-Powered-By removed, server_tokens off	OWASP A05
Payment Replay	HMAC-SHA256 + timing-safe compare + idempotency table	OWASP A08
Data Integrity	Row-level locking, transaction timeouts, server-side recalculation	OWASP A08
Audit Trail	AdminLog table, PM2 logs, CSP reports	OWASP A09
Session Management	httpOnly, secure, sameSite=lax, 30-day maxAge	OWASP A07
Error Handling	Production: errors-only logging, no stack traces to client	OWASP A09
Dependency Security	glob override for known vuln, Dependabot alerts	OWASP A06

3.3 Network Diagram



4. Technology Stack Summary

4.1 Backend

Component	Technology	Version
Runtime	Node.js	20.20.0
Framework	Next.js (App Router)	14.2.x
Language	TypeScript	5.3.x

Database	PostgreSQL	16.11
ORM	Prisma	5.10.x
Authentication	NextAuth.js	4.24.x
Validation	Zod	3.25.x
Password Hashing	bcryptjs	2.4.3
Telegram Bot	Telegraf	4.16.3
Excel Reports	ExcelJS	4.4.0
QR Generation	qrcode	1.5.3
XML Parsing	fast-xml-parser	5.3.3

4.2 Frontend

Component	Technology	Version
UI Framework	React	18.2.x
CSS Framework	Tailwind CSS	3.4.x
Component Library	shadcn/ui (Radix UI primitives)	Various
Maps	Leaflet + react-leaflet	1.9.4 / 4.2.1
Charts	Recharts	3.7.0
Icons	Lucide React	0.344.x
Date Utilities	date-fns	4.1.x
CSV Parsing	PapaParse	5.4.1

4.3 Infrastructure

Component	Technology	Details
Cloud	AWS EC2	t2.micro, us-east-1
OS	Ubuntu	22.04 LTS
Reverse Proxy	Nginx	With rate limiting
Process Manager	PM2	Cluster mode
CDN / WAF	Cloudflare	Free plan, Full Strict SSL
DNS	Cloudflare	Proxied A record
Monitoring	PM2 logs + Cloudflare Analytics	Error/access logs
PWA	Service Worker	Network-first, v2 cache

4.4 External Services

Service	Provider	Purpose	Protocol
Payment Gateway	TeleBirr (Ethio Telecom)	Mobile money payments	HTTPS REST, HMAC-SHA256
SMS Gateway	Negarit / GeezSMS	Ticket delivery, booking, reminders	HTTPS REST, Bearer token
Messaging	Telegram Bot API	Bot-based booking, tracking	HTTPS Webhook
Project Mgmt	ClickUp	Support tickets, alerts	HTTPS REST, API token
Map Tiles	OpenStreetMap	Map rendering	HTTPS (public CDN)
Routing	OSRM	Road route geometry	HTTPS (public API)

Geocoding	Nominatim	Reverse geocoding	HTTPS (public API)
Analytics	Cloudflare Web Analytics	Traffic analytics	Client-side beacon

5. Service Worker (PWA)

Property	Value
File	/public/sw.js
Cache Name	i-ticket-v2
Strategy	Network-first, cache fallback
Scope	Same-origin only (cross-origin requests skipped)
API Requests	Always network (never cached)
POST/PUT/DELETE	Never cached
Install	skipWaiting() (immediate activation)
Activate	clients.claim() + old cache purge

6. Process Manager Configuration

Property	Value
Manager	PM2
App Name	i-ticket
Exec Mode	Cluster
Instances	1 (limited by t2.micro 1GB RAM)
Max Memory	800 MB (restart threshold)
Auto-Restart	Yes
Max Restarts	10
Min Uptime	10 seconds
Kill Timeout	5000ms (graceful shutdown)
Cron Restart	Daily at 03:00 UTC (memory management)
Error Log	/var/log/pm2/i-ticket-error.log
Output Log	/var/log/pm2/i-ticket-out.log
Log Format	YYYY-MM-DD HH:mm:ss Z

End of Document 4.2.1b - System Architecture Diagram



DOCUMENT

4.2.1c

Entity Relationship Diagram

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Entity Relationship Diagram (ERD)

Document: 4.2.1c - Entity Relationship Diagram Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform
 URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Database Overview

Property	Value
Database Engine	PostgreSQL 16.11
ORM	Prisma 5.10.x
Total Models	39
ID Strategy	CUID (collision-resistant unique identifiers)
Timestamps	<code>createdAt</code> / <code>updatedAt</code> on all models

2. Entity Definitions

Fields marked with [SENSITIVE] require encryption or access control per INSA requirements. Fields marked with [PK] are primary keys, [FK] are foreign keys, [UQ] are unique constraints.

2.1 Core Business Entities

User

All system users: customers, company admins, staff, guest users

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
name	String?	Nullable (guest users)	PII
phone	String	[UQ] Unique	[SENSITIVE] PII, auth identifier
email	String?		[SENSITIVE] PII
password	String?	Nullable (guest users)	[SENSITIVE] bcrypt hash (12 rounds)
nationalId	String?		[SENSITIVE] Government ID
role	String	Default: "CUSTOMER"	Access control
companyId	String?	[FK] -> Company.id	Data segregation
staffRole	String?	ADMIN/DRIVER/CONDUCTOR/MANUAL_TICKETER/MECHANIC/FINANCE	Access control
staffStatus	String?	Default: "AVAILABLE"	
licenseNumber	String?		PII
employeeId	String?		
nextOfKinName	String?		[SENSITIVE] PII
nextOfKinPhone	String?		[SENSITIVE] PII
profilePicture	String?	URL/path	

isGuestUser	Boolean	Default: false	
mustChangePassword	Boolean	Default: false	Auth control
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Indexes: [role], [companyId], [isGuestUser, createdAt], [companyId, staffRole], [phone, isGuestUser], [mustChangePassword]

Company

Bus companies with complete data isolation

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
name	String		
logo	String?	URL/path	
phones	String	Default: "[] (JSON)	PII
email	String		[SENSITIVE] PII
isActive	Boolean	Default: true	
disableAutoHaltGlobally	Boolean	Default: false	
fax	String?		
website	String?		
address	String?		
poBox	String?		
tinNumber	String?		[SENSITIVE] Tax ID
bankName	String?		[SENSITIVE] Financial
bankAccount	String?		[SENSITIVE] Financial
bankBranch	String?		[SENSITIVE] Financial
adminName	String?		PII
adminPhone	String?		[SENSITIVE] PII
adminEmail	String?		[SENSITIVE] PII
supportName	String?		PII
supportPhone	String?		PII
preparedBy	String?		
reviewedBy	String?		
approvedBy	String?		
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Trip

Bus trips with full lifecycle management

Field	Type	Constraints	Sensitivity
-------	------	-------------	-------------

id	String	[PK] CUID	
companyId	String	[FK] -> Company.id	Data segregation
driverId	String?	[FK] -> User.id	
conductorId	String?	[FK] -> User.id	
manualTicketerId	String?	[FK] -> User.id	
vehicleId	String?	[FK] -> Vehicle.id	
origin	String		
destination	String		
route	String?		
intermediateStops	String?	JSON	
defaultPickup	String?		
defaultDropoff	String?		
departureTime	DateTime		
estimatedDuration	Int	Minutes	
actualDepartureTime	DateTime?		
actualArrivalTime	DateTime?		
distance	Int?	Kilometers	
price	Float		Financial
busType	String		
totalSlots	Int		
availableSlots	Int		
hasWater	Boolean	Default: false	
hasFood	Boolean	Default: false	
isActive	Boolean	Default: true	
status	String	Default: "SCHEDULED"	
delayReason	String?		
delayedAt	DateTime?		
lowSlotAlertSent	Boolean	Default: false	
bookingHalted	Boolean	Default: false	
adminResumedFromAutoHalt	Boolean	Default: false	
autoResumeEnabled	Boolean	Default: false	
reportGenerated	Boolean	Default: false	
version	Int	Default: 0 (optimistic lock)	
noShowCount	Int	Default: 0	
releasedSeats	Int	Default: 0	
replacementsSold	Int	Default: 0	
trackingActive	Boolean	Default: false	
trackingToken	String?	[UQ] Unique	[SENSITIVE] Auth token
lastLatitude	Float?		Location data
lastLongitude	Float?		Location data

lastSpeed	Float?		
lastPositionAt	DateTime?		
estimatedArrival	DateTime?		
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Indexes: [origin, destination, departureTime], [departureTime], [companyId], [driverId], [conductorId], [manualTicketerId], [vehicleId], [status], [companyId, status], [companyId, status, departureTime], [trackingActive, status], [version]

Booking

Customer booking records

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
tripId	String	[FK] -> Trip.id	
userId	String	[FK] -> User.id	
totalAmount	Float		[SENSITIVE] Financial
commission	Float	5% of totalAmount	[SENSITIVE] Financial
commissionVAT	Float	Default: 0 (15% of commission)	[SENSITIVE] Financial
status	String	Default: "PENDING"	
isQuickTicket	Boolean	Default: false	
isReplacement	Boolean	Default: false	
replacedPassengerId	String?	Audit trail link	
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Indexes: [userId, status], [tripId], [createdAt], [status, createdAt], [tripId, status]

Passenger

Individual passengers per booking

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
bookingId	String	[FK] -> Booking.id (CASCADE)	
name	String		[SENSITIVE] PII
nationalId	String		[SENSITIVE] Government ID
phone	String		[SENSITIVE] PII
seatNumber	Int?		
specialNeeds	String?		[SENSITIVE] Health data
pickupLocation	String?		
dropoffLocation	String?		
boardingStatus	String	Default: "PENDING"	

Indexes: [bookingId] , [bookingId, boardingStatus]

Ticket

Generated tickets with QR codes and verification codes

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
bookingId	String	[FK] -> Booking.id (CASCADE)	
tripId	String	[FK] -> Trip.id	
passengerName	String		PII
seatNumber	Int?		
qrCode	String	Data URL (base64 PNG)	[SENSITIVE] Auth artifact
shortCode	String	[UQ] Unique (6-char)	[SENSITIVE] Auth token
isUsed	Boolean	Default: false	
usedAt	DateTime?		
createdAt	DateTime	Auto	

Indexes: [tripId] , [shortCode] , [shortCode, isUsed]

Payment

Payment transaction records

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
bookingId	String	[UQ] [FK] -> Booking.id (CASCADE)	
amount	Float		[SENSITIVE] Financial
method	String	TELEBIRR / DEMO	
transactionId	String?		[SENSITIVE] Financial reference
status	String	Default: "PENDING"	
initiatedVia	String	Default: "WEB" (WEB/SMS)	
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Indexes: [status] , [initiatedVia, status] , [createdAt]

City

Shared city reference data (only shared resource across companies)

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
name	String	[UQ] Unique	
region	String?		

isActive	Boolean	Default: true	
tripCount	Int	Default: 0	
latitude	Float?		
longitude	Float?		
timezone	String?		
createdAt	DateTime	Auto	

Indexes: [name], [latitude, longitude]

2.2 Security & Audit Entities

AdminLog

Comprehensive audit trail for all sensitive operations

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
userId	String		Audit
action	String		[SENSITIVE] Audit trail
details	String?		[SENSITIVE] May contain PII
tripId	String?		
companyId	String?		
createdAt	DateTime	Auto	

Indexes: [userId, createdAt], [action], [createdAt], [companyId, createdAt]

ProcessedCallback

Payment callback idempotency and replay prevention

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
transactionId	String	[UQ] Unique	[SENSITIVE] Financial
bookingId	String		
callbackHash	String	[UQ] Unique (SHA-256)	[SENSITIVE] Security
nonce	String?		Security
status	String		
processedAt	DateTime	Auto	
amount	Float		[SENSITIVE] Financial
signature	String		[SENSITIVE] HMAC signature
rawPayload	Text		[SENSITIVE] Full callback data

Indexes: [transactionId], [bookingId], [callbackHash], [processedAt]

PasswordReset

Password reset token management

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
userId	String	[FK] -> User.id (CASCADE)	
tokenHash	String	[UQ] Unique	[SENSITIVE] bcrypt-hashed auth token
expiresAt	DateTime	1-hour TTL	
isUsed	Boolean	Default: false	
usedAt	DateTime?		
createdAt	DateTime	Auto	

Indexes: [userId, isUsed], [tokenHash, isUsed], [expiresAt]

2.3 Fleet Management Entities

Vehicle

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
companyId	String	[FK] -> Company.id (CASCADE)	Data segregation
plateNumber	String	[UQ] per company	
sideNumber	String?	[UQ] per company	
make	String		
model	String		
year	Int		
busType	String	MINI/STANDARD/LUXURY	
color	String?		
totalSeats	Int		
status	String	Default: "ACTIVE"	
registrationExpiry	DateTime?		
insuranceExpiry	DateTime?		
lastServiceDate	DateTime?		
nextServiceDate	DateTime?		
currentOdometer	Int?		
engineHours	Int?		
fuelCapacity	Int?		
fuelType	String?		
utilizationRate	Float?		
costPerKm	Float?		Financial
revenuePerKm	Float?		Financial
fuelCostMTD	Float	Default: 0	Financial

fuelCostYTD	Float	Default: 0	Financial
maintenanceCostMTD	Float	Default: 0	Financial
maintenanceCostYTD	Float	Default: 0	Financial
maintenanceRiskScore	Int?	0-100 (AI)	
predictedFailureDate	DateTime?	AI prediction	
predictedFailureType	String?		
purchasePrice	Float?		[SENSITIVE] Financial
purchaseDate	DateTime?		
lastLatitude	Float?		Location data
lastLongitude	Float?		Location data
lastPositionAt	DateTime?		
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Unique Constraints: [companyId, plateNumber], [companyId, sideNumber] **Indexes:** [companyId], [status], [plateNumber], [sideNumber], [maintenanceRiskScore], [inspectionDueDate], [predictedFailureDate], [currentOdometer]

WorkOrder

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
workOrderNumber	String	[UQ] WO-XXXXXX	
vehicleId	String	[FK] -> Vehicle.id	
companyId	String		Data segregation
title	String		
description	Text?		
taskType	String	PREVENTIVE/CORRECTIVE/INSPECTION/EMERGENCY	
priority	Int	Default: 2	
assignedStaffIds	Text?	JSON array of user IDs	
serviceProvider	String?	External shop	
status	String	Default: "OPEN"	
scheduledDate	DateTime?		
startedAt	DateTime?		
completedAt	DateTime?		
odometerAtService	Int?		
laborCost	Float	Default: 0	[SENSITIVE] Financial
partsCost	Float	Default: 0	[SENSITIVE] Financial
totalCost	Float	Default: 0	[SENSITIVE] Financial
completionNotes	Text?		
mechanicSignature	String?		PII
fixedOnTheJob	Boolean	Default: false	

fixedByStaffName	String?		PII
externalGarageName	String?		
externalGarageCost	Float?		Financial
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

Indexes: [vehicleId, status], [workOrderNumber], [scheduledDate], [companyId], [status, priority], [fixedOnTheJob]

VehicleInspection, MaintenanceSchedule, FuelEntry, OdometerLog, TripLog, WorkOrderPart, VehicleRiskHistory, VehicleDowntime

(Detailed field definitions available in Prisma schema - these support fleet management with company segregation via vehicleId/companyId foreign keys. Financial fields in FuelEntry (costBirr, costPerLiter) and WorkOrderPart (unitPrice, totalPrice) are SENSITIVE.)

2.4 GPS Tracking Entity

TripPosition

GPS breadcrumbs from driver phones

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
tripId	String	[FK] -> Trip.id (CASCADE)	
vehicleId	String?		
latitude	Float		[SENSITIVE] Location tracking
longitude	Float		[SENSITIVE] Location tracking
altitude	Float?		
accuracy	Float?	GPS accuracy (meters)	
heading	Float?	0-360 degrees	
speed	Float?	km/h	
recordedAt	DateTime	Device timestamp	
receivedAt	DateTime	Server timestamp	

Indexes: [tripId, receivedAt], [vehicleId, receivedAt] Retention: Auto-purged after 7 days for COMPLETED/CANCELLED trips

2.5 Session Entities

TelegramSession

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
chatId	BigInt	[UQ] Unique	[SENSITIVE] Telegram user ID
userId	String?	[FK] -> User.id	
sessionId	String	[UQ] Unique	
state	String	Default: "IDLE"	
language	String	Default: "EN"	

origin	String?		
destination	String?		
date	DateTime?		
selectedTripId	String?		
passengerCount	Int	Default: 0	
passengerData	String?	JSON	[SENSITIVE] PII in JSON
selectedSeats	String?	JSON	
bookingId	String?		
paymentTransactionId	String?		[SENSITIVE] Financial
expiresAt	DateTime	30-min TTL	
createdAt	DateTime	Auto	
updatedAt	DateTime	Auto	

SmsSession

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
phone	String		[SENSITIVE] PII
sessionId	String	[UQ] Unique	
state	String		
language	String	Default: "EN"	
passengerData	String?	JSON	[SENSITIVE] PII in JSON
expiresAt	DateTime	15-min TTL	

2.6 Sales & Financial Entities

SalesPerson

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
name	String		[SENSITIVE] PII
phone	String	[UQ] Unique	[SENSITIVE] PII
email	String?		[SENSITIVE] PII
password	String		[SENSITIVE] bcrypt hash
referralCode	String	[UQ] Unique	
status	String	Default: "ACTIVE"	
recruiterId	String?	[FK] -> SalesPerson.id (self-ref)	
tier	Int	Default: 1	
bankAccountName	String?		[SENSITIVE] Financial PII
bankAccountNumber	String?		[SENSITIVE] Financial
bankName	String?		[SENSITIVE] Financial
alternativePhone	String?		[SENSITIVE] PII

profilePicture	String?		
lastLoginAt	DateTime?		

SalesCommission

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
salesPersonId	String	[FK] -> SalesPerson.id (CASCADE)	
bookingId	String	[UQ] [FK] -> Booking.id (CASCADE)	
ticketAmount	Float		[SENSITIVE] Financial
platformCommission	Float		[SENSITIVE] Financial
salesCommission	Float		[SENSITIVE] Financial
recruiterCommissionAmount	Float	Default: 0	[SENSITIVE] Financial
recruiterId	String?	[FK] -> SalesPerson.id	
status	String	Default: "PENDING"	
payoutId	String?	[FK] -> SalesPayout.id	
paidAt	DateTime?		

2.7 Communication Entities

SupportTicket

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
ticketNumber	String	[UQ] SUP-XXXXXX	
name	String		[SENSITIVE] PII
email	String		[SENSITIVE] PII
phone	String?		[SENSITIVE] PII
subject	String		
message	Text		May contain PII
category	String	Default: "GENERAL"	
priority	Int	Default: 2	
status	String	Default: "OPEN"	
resolution	Text?		
userAgent	String?		[SENSITIVE] Device fingerprint
ipAddress	String?		[SENSITIVE] Network identifier

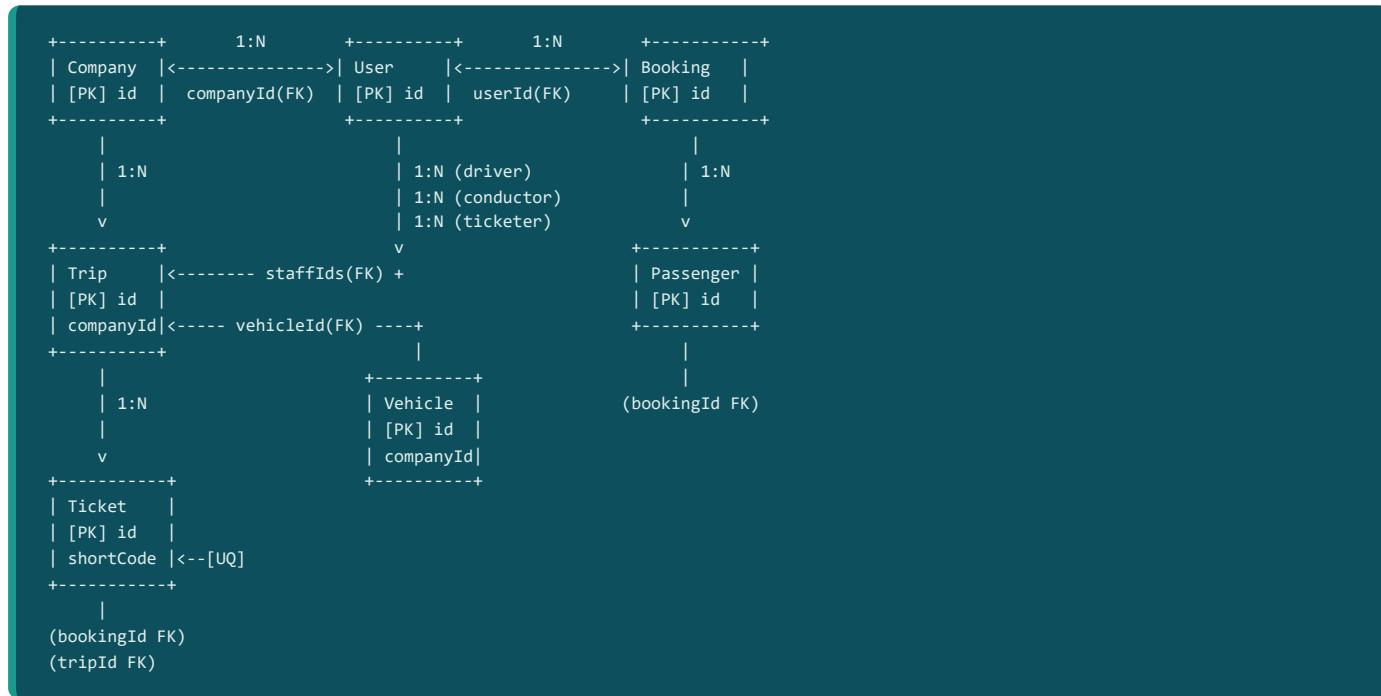
PlatformStaff

Field	Type	Constraints	Sensitivity
id	String	[PK] CUID	
userId	String	[UQ] [FK] -> User.id (CASCADE)	
role	String		Access control

department	String		
employeeld	String	[UQ] iTKT-XXX	[SENSITIVE] HR data
email	String	[UQ]	[SENSITIVE] PII
phone	String		[SENSITIVE] PII
position	String		HR data
hireDate	DateTime		HR data
status	String	Default: "ACTIVE"	
reportsTo	String?	Manager userId	
permissions	Text	Default: "{}" (JSON)	[SENSITIVE] Access control

3. Relationship Diagram

3.1 Core Business Relationships



3.2 Complete Relationship Table

Parent Entity	Child Entity	Relationship	Foreign Key	On Delete
Company	User	1:N	User.companyId	-
Company	Trip	1:N	Trip.companyId	-
Company	Vehicle	1:N	Vehicle.companyId	CASCADE
Company	TripTemplate	1:N	TripTemplate.companyId	CASCADE
Company	CompanyMessage	1:N	CompanyMessage.companyId	CASCADE
User	Booking	1:N	Booking.userId	-
User	Trip (driver)	1:N	Trip.driverId	SET NULL
User	Trip (conductor)	1:N	Trip.conductorId	SET NULL
User	Trip (ticketer)	1:N	Trip.manualTicketerId	SET NULL

User	PasswordReset	1:N	PasswordReset.userId	CASCADE
User	CompanyMessage	1:N	CompanyMessage.senderId	CASCADE
User	PlatformStaff	1:1	PlatformStaff.userId	CASCADE
User	TelegramSession	1:N	TelegramSession.userId	SET NULL
User	SalesReferral	1:1	SalesReferral.userId	CASCADE
Trip	Booking	1:N	Booking.tripId	-
Trip	Ticket	1:N	Ticket.tripId	-
Trip	TripLog	1:1	TripLog.tripId	CASCADE
Trip	TripPosition	1:N	TripPosition.tripId	CASCADE
Trip	ManifestDownload	1:N	ManifestDownload.tripId	CASCADE
Vehicle	Trip	1:N	Trip.vehicleId	SET NULL
Vehicle	MaintenanceSchedule	1:N	MaintenanceSchedule.vehicleId	CASCADE
Vehicle	WorkOrder	1:N	WorkOrder.vehicleId	-
Vehicle	FuelEntry	1:N	FuelEntry.vehicleId	-
Vehicle	VehicleInspection	1:N	VehicleInspection.vehicleId	-
Vehicle	OdometerLog	1:N	OdometerLog.vehicleId	CASCADE
Vehicle	TripLog	1:N	TripLog.vehicleId	-
Vehicle	VehicleRiskHistory	1:N	VehicleRiskHistory.vehicleId	CASCADE
Vehicle	VehicleDowntime	1:N	VehicleDowntime.vehicleId	CASCADE
Booking	Passenger	1:N	Passenger.bookingId	CASCADE
Booking	Ticket	1:N	Ticket.bookingId	CASCADE
Booking	Payment	1:1	Payment.bookingId	CASCADE
Booking	SalesCommission	1:1	SalesCommission.bookingId	CASCADE
WorkOrder	WorkOrderPart	1:N	WorkOrderPart.workOrderId	CASCADE
WorkOrder	WorkOrderMessage	1:N	WorkOrderMessage.workOrderId	CASCADE
TripMessage	TripMessageReadReceipt	1:N	ReadReceipt.messageId	CASCADE
WorkOrderMessage	WorkOrderMessageReadReceipt	1:N	ReadReceipt.messageId	CASCADE
Notification	Notification (children)	1:N (self-ref)	Notification.parentId	CASCADE
SalesPerson	SalesPerson (recruits)	1:N (self-ref)	SalesPerson.recruiterId	SET NULL
SalesPerson	SalesReferral	1:N	SalesReferral.salesPersonId	CASCADE
SalesPerson	SalesCommission	1:N	SalesCommission.salesPersonId	CASCADE
SalesPerson	SalesCommission (recruiter)	1:N	SalesCommission.recruiterId	SET NULL
SalesPerson	SalesQrScan	1:N	SalesQrScan.salesPersonId	CASCADE
SalesPerson	SalesPayout	1:N	SalesPayout.salesPersonId	CASCADE
SalesPayout	SalesCommission	1:N	SalesCommission.payoutId	-

3.3 Full ERD Diagram





Legend:

[PK] = Primary Key [S] = SENSITIVE field
[FK] = Foreign Key [UQ] = Unique constraint
1:N = One-to-Many 1:1 = One-to-One

4. Sensitive Fields Summary

4.1 Fields Requiring Encryption at Rest

Entity	Field	Data Type	Current Protection	Recommendation
User	password	String	bcrypt hash (12 rounds)	Adequate - hashed, not encrypted
User	nationalId	String	Plaintext	Consider AES-256 encryption
User	phone	String	Plaintext	Consider AES-256 encryption
User	email	String	Plaintext	Consider AES-256 encryption
Passenger	nationalId	String	Plaintext	Consider AES-256 encryption
Passenger	phone	String	Plaintext	Consider AES-256 encryption
SalesPerson	password	String	bcrypt hash	Adequate
SalesPerson	bankAccountNumber	String	Plaintext	Requires AES-256 encryption
Company	bankAccount	String	Plaintext	Requires AES-256 encryption
Company	tinNumber	String	Plaintext	Consider AES-256 encryption
PasswordReset	tokenHash	String	bcrypt hash	Adequate
ProcessedCallback	rawPayload	Text	Plaintext	Consider encryption
Ticket	shortCode	String	Plaintext (6-char)	Adequate (short-lived, single-use)
Trip	trackingToken	String	Plaintext	Consider encryption

4.2 Fields Requiring Access Control

Entity	Field	Access Rule
--------	-------	-------------

User	password	Never exposed in API responses
User	nationalId	Only visible to company admin (same company) and super admin
Passenger	nationalId	Only visible to trip staff and company admin
Passenger	phone	Only visible to trip staff and company admin
Company	bankAccount	Only visible to company admin and super admin
Company	tinNumber	Only visible to company admin and super admin
SalesPerson	bankAccountNumber	Only visible to sales person (self) and super admin
AdminLog	details	Only visible to super admin and company admin (company-scoped)
ProcessedCallback	rawPayload	Only visible to super admin
PlatformStaff	permissions	Only visible to super admin
TripPosition	latitude/longitude	Public for DEPARTED trips (passenger tracking)
SupportTicket	ipAddress	Only visible to super admin

4.3 Data Retention Policies

Entity	Retention	Mechanism
TripPosition	7 days after trip COMPLETED/CANCELLED	Cron job auto-purge
SmsSession	15 minutes	Cron job cleanup
TelegramSession	30 minutes	Cron job cleanup
PasswordReset	1 hour (token expiry)	isUsed flag + expiresAt
Payment (PENDING)	10 minutes	Cron job cancellation
Booking (PENDING)	10 minutes	Cron job cancellation

5. Unique Constraints Summary

Entity	Field(s)	Purpose
User	phone	Primary auth identifier
City	name	Prevent duplicate cities
Ticket	shortCode	Ticket verification token
Payment	bookingId	One payment per booking
Trip	trackingToken	OsmAnd GPS auth
ProcessedCallback	transactionId	Payment replay prevention
ProcessedCallback	callbackHash	Duplicate callback prevention
PasswordReset	tokenHash	Token uniqueness
SalesPerson	phone	Auth identifier
SalesPerson	referralCode	Referral attribution
SalesReferral	userId	One referral per user
SalesCommission	bookingId	One commission per booking
TelegramSession	chatId	One session per Telegram user
TelegramSession	sessionId	Session uniqueness
SmsSession	sessionId	Session uniqueness

PlatformStaff	userId	One staff record per user
PlatformStaff	employeeId	Employee ID uniqueness
PlatformStaff	email	Email uniqueness
WorkOrder	workOrderNumber	WO reference uniqueness
SupportTicket	ticketNumber	Ticket reference uniqueness
Vehicle	[companyId, plateNumber]	Plate unique per company
Vehicle	[companyId, sideNumber]	Side number unique per company
TripLog	tripId	One log per trip
TripMessageReadReceipt	[messageId, userId]	One read per user per message
WorkOrderMessageReadReceipt	[messageId, userId]	One read per user per message

6. Index Summary

Total indexes across all models: 85+

Key performance indexes:

- **Trip queries:** [companyId, status, departureTime] (compound)
- **Booking lookups:** [tripId, status], [userId, status]
- **Ticket verification:** [shortCode, isUsed]
- **Fleet tracking:** [trackingActive, status]
- **Audit trail:** [userId, createdAt], [companyId, createdAt]
- **Risk scoring:** [maintenanceRiskScore], [predictedFailureDate]

End of Document 4.2.1c - Entity Relationship Diagram



D O C U M E N T

4.2.2

Features of Web Application

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Features of the Web Application

Document: 4.2.2 - Features of the Web Application Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Development Framework

Property	Value
Framework	Next.js 14 (App Router)
Language	TypeScript 5.3.x
UI Library	React 18.2.x
Runtime	Node.js 20.20.0
Database	PostgreSQL 16.11
ORM	Prisma 5.10.x
Authentication	NextAuth.js 4.24.x (Credentials Provider, JWT strategy)
CSS Framework	Tailwind CSS 3.4.x
Component Library	shadcn/ui (built on Radix UI primitives)
Package Manager	npm
Build Tool	Next.js built-in (SWC compiler)

2. Libraries and Plugins Integrated

2.1 Production Dependencies

Library	Version	Purpose	Category
next	^14.2.0	Full-stack web framework (SSR, API routes, middleware)	Core
react	^18.2.0	UI rendering library	Core
react-dom	^18.2.0	React DOM bindings	Core
next-auth	^4.24.0	Authentication (JWT sessions, credentials provider)	Auth
@prisma/client	^5.10.0	Database ORM (PostgreSQL, parameterized queries)	Database
zod	^3.25.76	Runtime schema validation (all API inputs)	Security
bcryptjs	^2.4.3	Password hashing (12 salt rounds)	Security
telegraf	^4.16.3	Telegram Bot API framework	Integration
leaflet	^1.9.4	Interactive map rendering (GPS tracking)	Maps
react-leaflet	^4.2.1	React bindings for Leaflet (v4 for React 18)	Maps
recharts	^3.7.0	Chart/graph rendering (fleet analytics dashboards)	Visualization
exceljs	^4.4.0	Excel file generation (reports, manifests)	Reports
qrcode	^1.5.3	QR code generation (ticket verification)	Ticketing
date-fns	^4.1.0	Date/time manipulation and formatting	Utility

<code>fast-xml-parser</code>	^5.3.3	XML parsing (TeleBirr API responses)	Integration
<code>html2canvas</code>	^1.4.1	HTML-to-image conversion (ticket screenshots)	Utility
<code>papaparse</code>	^5.4.1	CSV parsing (trip import)	Utility
<code>canvas-confetti</code>	^1.9.4	Confetti animation (booking success)	UI
<code>cmdk</code>	^1.1.1	Command palette component	UI
<code>sonner</code>	^2.0.7	Toast notification system	UI
<code>lucide-react</code>	^0.344.0	Icon library (200+ icons)	UI
<code>class-variance-authority</code>	^0.7.0	CSS variant utility	UI
<code>clsx</code>	^2.1.0	Conditional CSS class merging	UI
<code>tailwind-merge</code>	^2.2.0	Tailwind class conflict resolution	UI
<code>@radix-ui/react-*</code>	Various	15+ headless UI primitives (dialog, dropdown, tabs, etc.)	UI

2.2 Development Dependencies

Library	Version	Purpose
<code>typescript</code>	^5.3.0	Type-safe development
<code>prisma</code>	^5.10.0	Database schema management, migrations
<code>eslint</code>	^8.56.0	Code linting
<code>eslint-config-next</code>	^14.2.0	Next.js-specific lint rules
<code>tailwindcss</code>	^3.4.0	CSS framework (build-time)
<code>postcss</code>	^8.4.0	CSS processing
<code>autoprefixer</code>	^10.4.0	CSS vendor prefixing
<code>sharp</code>	^0.34.5	Image optimization (Next.js Image component)
<code>tsx</code>	^4.7.0	TypeScript script execution
<code>dotenv</code>	^16.4.5	Environment variable loading
<code>playwright</code>	^1.58.1	End-to-end browser testing

2.3 Dependency Security

Measure	Details
Dependency override	<code>glob: ^10.5.0</code> (patched known vulnerability)
Vulnerability monitoring	GitHub Dependabot alerts enabled
Resolved vulnerabilities	6 of 7 Dependabot alerts resolved (v2.13.0)

3. Custom-Developed Modules

3.1 Business Logic Modules (`src/lib/`)

Module	File(s)	Description
Authentication	<code>auth.ts</code>	NextAuth config with in-memory rate limiting (per-phone + per-IP), bcrypt password handling, JWT token management, production secret validation

Database Client	db.ts	Prisma singleton with <code>transactionWithTimeout()</code> (10s timeout, dual Promise.race + Prisma native timeout)
Rate Limiter	rate-limit.ts	In-memory rate limiter with per-endpoint configs, 100K max store, emergency cleanup at 80% capacity, auto-cleanups every 60s
Commission Engine	commission.ts	5% platform commission + 15% VAT calculation, TeleBirr fee (0.5%), net revenue calculation, multi-tier sales commission (70/30 recruiter split)
Timezone Utilities	utils.ts	Ethiopia timezone helpers (<code>hasDepartedEthiopia()</code> , <code>isTodayEthiopia()</code> , <code>isSameDayEthiopia()</code>) to prevent UTC/local time mismatches on AWS
Fuzzy Search	fuzzy-match.ts	Levenshtein distance-based fuzzy matching for city names and route stops
Payment - TeleBirr	payments/telebirr.ts	TeleBirr API client, HMAC-SHA256 request signing, signature verification with <code>crypto.timingSafeEqual()</code> , nonce generation
Payment - Callback Hash	payments/callback-hash.ts	SHA-256 callback deduplication, replay prevention
SMS Gateway	sms/gateway.ts	Outbound SMS (ticket delivery, reminders, timeout notifications), inbound SMS webhook processing with HMAC verification
Telegram Bot	telegram/bot.ts	Full Telegram bot with bilingual support (EN/AM), booking wizard, seat selection, ticket delivery, bus tracking
GPS Tracking	tracking/update-position.ts	Shared GPS position processing for web and OsmAnd inputs, ETA calculation (Haversine + 1.3x winding factor)
Audio Keep-Alive	tracking/audio-keep-alive.ts	Silent audio playback to prevent browser JS suspension during background GPS tracking
Predictive Maintenance AI	ai/predictive-maintenance.ts	Algorithmic risk scoring (0-100) based on odometer, service history, defects, fuel efficiency, age/mileage
ClickUp Integration	clickup/client.ts	Support ticket creation, low-slot alerts, audit log task creation in ClickUp
Password Reset	password-reset.ts	Crypto-random token generation, bcrypt token hashing, 1-hour expiry, single-use enforcement

3.2 API Route Modules (`src/app/api/`)

Category	Routes	Auth Required	Description
Auth	<code>/api/auth/[...nextauth]</code> , <code>/api/auth/register</code> , <code>/api/auth/force-change-password</code> , <code>/api/auth/forgot-password</code> , <code>/api/auth/reset-password</code>	Varies	Registration, login, password management
Trips	<code>/api/trips</code>	No (public)	Trip search with origin, destination, date filters
Bookings	<code>/api/bookings</code>	Optional (guest allowed)	Create bookings with seat selection, multi-passenger support
Payments	<code>/api/payments</code> , <code>/api/payments/telebirr/callback</code>	Yes / HMAC	Payment initiation (Demo/TeleBirr), callback processing with 5-layer security
Tickets	<code>/api/tickets/verify</code>	Yes (staff)	QR code ticket verification, auto boarding status update
Tracking - Public	<code>/api/track/[code]</code>	No	Public booking/ticket tracking by shortCode
Tracking - GPS	<code>/api/tracking/update</code> , <code>/api/tracking/osmand</code> , <code>/api/tracking/[tripId]</code> , <code>/api/tracking/fleet</code> , <code>/api/tracking/generate-token</code> , <code>/api/tracking/active-trip</code>	Varies	Driver GPS submission, passenger tracking, fleet map, OsmAnd token generation
Company - Trips	<code>/api/company/trips/*</code>	Yes (company)	Trip CRUD, status transitions, boarding management



Company - No-Show	/api/company/trips/[tripId]/boarding-status , no-show , replacement-ticket	Yes (company)	No-show marking, replacement ticket sales
Company - Staff	/api/company/staff/*	Yes (company)	Staff CRUD, role management, status tracking
Company - Vehicles	/api/company/vehicles/*	Yes (company)	Vehicle CRUD, inspections, fuel entries, work orders
Company - Analytics	/api/company/analytics/fleet-health , risk-trends , failure-timeline , cost-forecast , vehicle-comparison , maintenance-windows , route-wear , compliance-calendar	Yes (company)	8 fleet analytics endpoints
Company - Reports	/api/company/reports/maintenance , maintenance/export , vehicle-tco , downtime , fleet-analytics/export	Yes (company)	5 report endpoints with Excel export
Admin	/api/admin/stats , companies , trips , bookings , manifests , audit-logs , platform-staff , sales-persons , tax-reports , analytics/*	Yes (super admin)	Full platform management
Cron	/api/cron/cleanup , trip-reminders , predictive-maintenance , telegram-cleanup	Bearer token	Scheduled job endpoints
Telegram	/api/telegram/webhook	Telegram-signed	Bot webhook handler
SMS	/api/sms/incoming	HMAC signature	Inbound SMS booking webhook
Support	/api/support/tickets	No (public)	Support ticket creation
Cities	/api/cities , /api/cities/coordinates	No (public)	City list and coordinates for maps
CSP	/api/csp-report	No	CSP violation report endpoint

3.3 Frontend Components

Category	Key Components	Description
Booking Flow	TripSearch, SeatSelector, PassengerForm, PaymentForm, BookingConfirmation	Multi-step booking wizard with real-time seat availability
GPS Tracking	DriverTrackingPage, PassengerTrackingMap, FleetMap, BusMarker, RouteOverlay, ETABadge, OsmAndSetup, GPSStatusIndicator	8 tracking components for driver/passenger/fleet views
Fleet Analytics	FleetHealthGauge, RiskDistribution, TrendChart, CostForecast, FailureTimeline, VehicleComparison, MaintenanceWindows, ComplianceCalendar, RouteWearAnalysis	10 chart components for predictive maintenance dashboard
No-Show Management	BoardingChecklist, ReplacementTicketCard	Boarding status management and replacement ticket sales
Route Selection	RouteStopCombobox, PickupMapModal, CityCombobox	Fuzzy search autocomplete with interactive map selection
Reports	MaintenanceCostReport, VehicleTCOResult, DowntimeReport, FleetAnalyticsExport	Tabbed report views with Excel export
Shared UI	15+ Radix-based shadcn/ui components (Button, Dialog, Select, Tabs, Table, Card, Badge, etc.)	Accessible, composable UI primitives

4. Third-Party Service Integrations

Service	Provider	Purpose	Protocol	Auth Method	Data Exchanged
TeleBirr	Ethio Telecom	Mobile money payment processing	HTTPS REST	HMAC-SHA256 signed requests	Payment initiation (phone, amount, merchant code) -> TeleBirr; Payment callback (transactionId, status, amount, signature) <- TeleBirr

SMS Gateway	Negarit / GeezSMS	SMS delivery for tickets, reminders, booking	HTTPS REST	Bearer token (API key)	Outbound: ticket codes, trip info, reminders; Inbound: booking commands via webhook (HMAC-SHA256 verified)
Telegram Bot API	Telegram	Bot-based booking, ticket delivery, bus tracking	HTTPS Webhook	Bot token	Messages, inline keyboards, location sharing, QR code images
ClickUp	ClickUp	Support ticket management, operational alerts	HTTPS REST	API token (Bearer)	Task creation for support tickets, low-slot alerts, audit entries
OpenStreetMap	OSM Foundation	Map tile rendering for GPS tracking UI	HTTPS CDN	None (public)	PNG map tile images
OSRM	Project OSRM	Road route geometry for map overlays	HTTPS REST	None (public)	Road coordinates array (JSON) for route line rendering
Nominatim	OSM Foundation	Reverse geocoding (coordinates -> address)	HTTPS REST	None (public)	Location name from latitude/longitude
Cloudflare	Cloudflare	CDN, WAF, DDoS protection, DNS, analytics	HTTPS Proxy	API token (for DNS management)	All web traffic proxied, SSL termination, Web Analytics beacon
QR Code Generator	qrcode npm library	QR code generation for tickets	Local (library)	N/A	Verification URL encoded as QR data URL (base64 PNG)

5. Actor / User Types

5.1 Authentication Roles

Role	Auth Method	Session	Access Level
Customer (Registered)	Phone + password	JWT cookie (30 days)	Search trips, book tickets, view tickets, track bus
Guest Customer	Phone number only (no password)	Payment = verification	Book tickets, view tickets (no account management)
Company Admin	Phone + password	JWT cookie (30 days)	Full company management (trips, staff, vehicles, reports)
Super Admin	Phone + password	JWT cookie (30 days)	Full platform access (all companies, system config, analytics)
Sales Person	Phone + password (separate table)	JWT cookie (30 days)	Referral dashboard, commission tracking, QR codes

5.2 Staff Sub-Roles (under Company Admin role)

Staff Role	Portal	Capabilities
ADMIN	/company/*	Full company management
DRIVER	/driver/*	GPS tracking, trip departure, odometer entry, inspections
CONDUCTOR	/company/*	Ticket verification, boarding management, trip messaging
MANUAL_TICKETER	/cashier/*	Manual ticket sales (cash), replacement ticket sales
MECHANIC	/mechanic/*	Work orders, vehicle inspections, field repairs
FINANCE	/finance/*	Financial reports, cost tracking, commission viewing

5.3 Platform Staff Roles (under Super Admin)

Department	Roles
Finance	Accountant, Financial Analyst, Billing Manager, Revenue Manager, Payroll Specialist



Operations	Operations Manager, Dispatch Coordinator, System Administrator, QA Specialist
Support	Support Agent, Support Team Lead, Customer Success Manager, Technical Support
Marketing	Marketing Manager, Content Manager, Business Analyst, Partnership Manager
Technical	Platform Admin, Database Admin, DevOps Engineer, Security Analyst
Compliance	Compliance Officer, Legal Advisor, Data Protection Officer
Management	Platform Admin (CEO), General Manager, Department Head

5.4 External System Actors

Actor	Interaction	Auth
TeleBirr API	Payment callbacks	HMAC-SHA256 signature
SMS Gateway	Inbound SMS webhooks	HMAC-SHA256 signature
Telegram API	Bot update webhooks	Telegram-signed
Cron Scheduler	Scheduled job triggers	Bearer token (CRON_SECRET)
OsmAnd App	Background GPS data	Trip tracking token

6. System Dependencies and Minimum Hosting Requirements

6.1 Server Requirements

Component	Minimum	Current Production
CPU	1 vCPU	1 vCPU (EC2 t2.micro)
RAM	1 GB	1 GB (PM2 limit: 800 MB)
Storage	10 GB SSD	7.6 GB EBS (81% used)
OS	Ubuntu 20.04+	Ubuntu 22.04 LTS
Node.js	18.x+	20.20.0 (LTS)
PostgreSQL	14+	16.11
Network	HTTPS (port 443)	Cloudflare -> Nginx -> Node.js

6.2 Software Dependencies

Component	Version	Required
Node.js	>= 18.0.0	Yes
npm	>= 9.0.0	Yes
PostgreSQL	>= 14.0	Yes
Nginx	>= 1.18	Yes (reverse proxy)
PM2	Latest	Yes (process manager)
Git	>= 2.0	Yes (deployment)

6.3 Environment Variables Required

Variable	Purpose	Sensitivity
DATABASE_URL	PostgreSQL connection string	HIGH - contains credentials



NEXTAUTH_SECRET	JWT signing key (>= 32 chars)	HIGH - cryptographic secret
NEXTAUTH_URL	Application base URL	LOW
TELEBIRR_APP_ID	TeleBirr merchant app ID	MEDIUM
TELEBIRR_APP_KEY	TeleBirr HMAC signing key	HIGH - cryptographic secret
TELEBIRR_API_URL	TeleBirr API endpoint	LOW
TELEBIRR_NOTIFY_URL	TeleBirr callback URL	LOW
TELEBIRR_MERCHANT_CODE	TeleBirr merchant identifier	MEDIUM
TELEBIRR_WEBHOOK_IPS	Allowed callback IPs	MEDIUM
SMS_GATEWAY_URL	SMS provider API endpoint	LOW
SMS_GATEWAY_API_KEY	SMS provider auth key	HIGH - API credential
SMS_GATEWAY_SHORTCODE	SMS shortcode	LOW
SMS_WEBHOOK_SECRET	Inbound SMS HMAC secret	HIGH - cryptographic secret
TELEGRAM_BOT_TOKEN	Telegram bot API token	HIGH - API credential
TELEGRAM_BOT_ENABLED	Bot feature flag	LOW
CLICKUP_API_KEY	ClickUp API token	MEDIUM
CLICKUP_ENABLED	ClickUp feature flag	LOW
CRON_SECRET	Cron endpoint auth token	HIGH - auth secret
DEMO_MODE	Payment simulation flag	LOW (blocked in production)

7. Implemented Security Standards and Policies

7.1 Standards Compliance

Standard	Implementation
OWASP Top 10 (2021)	Addressed across all categories (see mapping below)
NIST SP 800-63B	Password hashing with bcrypt (12 rounds), session management
NIST SP 800-52	TLS 1.2 minimum enforced via Cloudflare
ISO/IEC 27001	Access control (RBAC), audit logging, data classification

7.2 OWASP Top 10 Mapping

OWASP ID	Category	i-Ticket Mitigation
A01:2021	Broken Access Control	RBAC with 3 main roles + 6 staff sub-roles; company segregation enforced on every API (companyId filter); <code>SameSite=lax</code> cookies; <code>form-action 'self'</code> CSP
A02:2021	Cryptographic Failures	bcrypt passwords (12 rounds); HMAC-SHA256 payment signatures; TLS 1.2+ (Cloudflare Full Strict); HSTS preload; <code>upgrade-insecure-requests</code> CSP
A03:2021	Injection	Prisma ORM (parameterized queries - no raw SQL); Zod validation on all API inputs; React auto-escaping (XSS); CSP with <code>default-src 'self'</code>
A04:2021	Insecure Design	3-tier rate limiting (Cloudflare -> Nginx -> App); 10-minute payment timeout; <code>SELECT FOR UPDATE NOWAIT</code> for concurrent booking; server-side amount recalculations

A05:2021	Security Misconfiguration	X-Powered-By removed; server_tokens off ; X-Frame-Options: DENY ; X-Content-Type-Options: nosniff ; Referrer-Policy: strict-origin-when-cross-origin ; Permissions-Policy restricting sensors
A06:2021	Vulnerable Components	Dependabot alerts (6/7 resolved); glob override for known vuln; Regular dependency updates
A07:2021	Identification & Auth Failures	Rate-limited login (5/30min per phone, 10/15min per IP, 15min lockout); bcrypt (12 rounds); 32-char minimum secret; production secret validation
A08:2021	Software & Data Integrity Failures	HMAC-SHA256 + timing-safe compare for payment callbacks; SHA-256 callback hash for replay prevention; ProcessedCallback idempotency table; Server-side commission recalculation
A09:2021	Security Logging & Monitoring Failures	AdminLog table for all sensitive operations; CSP violation reporting to /api/csp-report ; PM2 error/output logs; Cloudflare Web Analytics
A10:2021	Server-Side Request Forgery	CSP connect-src whitelist (only specific domains); No user-controlled URL fetching in API routes

7.3 Application-Level Security Policies

Policy	Implementation
Password Policy	bcrypt hash (12 rounds), nullable for guest users, mustChangePassword flag for temp passwords
Session Policy	JWT, 30-day maxAge, httpOnly + secure + sameSite=lax cookies
Payment Security	10-min payment window, 5-gate callback verification (hash, replay, signature, timestamp, IP), server-side amount calculation
Data Isolation	Complete company segregation - every company API filters by companyId from session. Only shared resource: City database
Input Validation	Zod schemas on all API inputs; phone format ^09\d{8}\$; parseInt rejects scientific notation
Error Handling	Production: error-only logging, no stack traces in responses; Development: query + warn + error logging
Concurrency Control	SELECT FOR UPDATE NOWAIT row locking; optimistic locking (Trip.version); 10s transaction timeouts
Audit Trail	AdminLog for: login, booking, payment, status changes, no-show, replacement, staff changes, company activation
Data Retention	GPS positions: 7 days; SMS sessions: 15 min; Telegram sessions: 30 min; Password reset tokens: 1 hour
Guest Booking	Phone payment = verification (no OTP required). Business rule: not a security gap

8. Existing Security Infrastructure

Component	Type	Details
Cloudflare	CDN / WAF / DDoS	Free plan; SSL Full Strict; TLS 1.2 minimum; HSTS with preload; Automatic DDoS protection; Web Analytics
Nginx	Reverse Proxy / Rate Limiter	Rate limiting (100r/m general, 60r/m API); Server header stripping; Max body size 5 MB; Proxy timeouts 60s
Next.js Middleware	Application Firewall	CSP injection with strict policy; Cache-Control: no-store; CSP violation reporting
Application Rate Limiter	In-Memory Rate Limiter	Per-endpoint configs (login: 5/30min, register: 3/hr, booking: 10/min, payment: 3/hr); 100K max store; Auto-cleanups
PM2	Process Manager / Monitor	Auto-restart on crash; Memory limit (800 MB); Daily cron restart at 3 AM; Error/output logging
Service Worker	PWA / Cache	Network-first strategy; Skips cross-origin requests; Skips API requests; Version-based cache invalidation
CSP Reporting	Violation Monitor	/api/csp-report endpoint logs violations to PM2; Report-To + Reporting-Endpoints headers
AdminLog	Audit Trail	Database-backed audit log for all sensitive operations
ProcessedCallback	Payment Replay Guard	SHA-256 callback hash + unique transactionId for idempotency

8.1 Not Currently Implemented

Component	Status	Notes
IDS/IPS	Not implemented	Cloudflare provides basic bot/DDoS protection at edge
Load Balancer	Not needed	Single EC2 instance with PM2 (1 instance due to t2.micro)
VPN	Not implemented	SSH key-based access only
Database Encryption at Rest	Not implemented	PostgreSQL uses filesystem-level storage; no TDE configured
Field-Level Encryption	Not implemented	Sensitive fields (nationalId, bankAccount) stored plaintext - recommended for future
Web Application Firewall (dedicated)	Cloudflare WAF (basic)	Free plan provides basic WAF rules; no custom WAF rules
SIEM	Not implemented	Logs stored in PM2 + AdminLog table; no centralized SIEM
Backup System	Not documented	PostgreSQL backup schedule not formalized

End of Document 4.2.2 - Features of the Web Application





D O C U M E N T

4.2.3

Testing Scope

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Testing Scope Definition

Document: 4.2.3 - Define Specific Testing Scope Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform
URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Assets in Scope

#	Asset Name	URL / IP Address	Type	Description
1	Public Web Portal	https://i-ticket.et	Web Application	Customer-facing booking platform (search, book, pay, track)
2	Admin Dashboard	https://i-ticket.et/admin/*	Web Application	Super Admin platform management portal
3	Company Portal	https://i-ticket.et/company/*	Web Application	Bus company management (trips, staff, fleet, reports)
4	Driver Portal	https://i-ticket.et/driver/*	Web Application	Driver GPS tracking page
5	Cashier Portal	https://i-ticket.et/cashier/*	Web Application	Manual ticket sales portal
6	Mechanic Portal	https://i-ticket.et/mechanic/*	Web Application	Vehicle inspection and work order portal
7	Finance Portal	https://i-ticket.et/finance/*	Web Application	Financial reports portal
8	Sales Portal	https://i-ticket.et/sales/*	Web Application	Sales referral and commission portal
9	REST API	https://i-ticket.et/api/*	API	All backend API endpoints (see Section 3)
10	Telegram Bot	https://t.me/i_ticket_busBot	Bot	Bilingual booking bot (@i_ticket_busBot)
11	Telegram Webhook	https://i-ticket.et/api/telegram/webhook	API Webhook	Telegram bot update receiver
12	SMS Webhook	https://i-ticket.et/api/sms/incoming	API Webhook	Inbound SMS booking receiver
13	TeleBirr Callback	https://i-ticket.et/api/payments/telebirr/callback	API Webhook	Payment callback endpoint
14	Origin Server	54.147.33.168	EC2 Instance	AWS EC2 t2.micro (Ubuntu 22.04)
15	Service Worker	https://i-ticket.et/sw.js	PWA	Progressive web app service worker
16	Manifest	https://i-ticket.et/manifest.json	PWA	Web app manifest for installable PWA

2. Test Account Credentials

2.1 Web Application Accounts

#	Role	Phone	Password	Portal URL	Capabilities
1	Super Admin	0911223344	demo123	/admin/*	Full platform access: all companies, system analytics, user management, audit logs, tax reports, platform staff
2	Company Admin (Selam Bus)	0922345678	demo123	/company/*	Full company management: trips, staff, vehicles, fleet analytics, reports, boarding management

3	Customer	0912345678	demo123	/(customer)/*	Search trips, book tickets, view tickets, track bus, manage profile
4	Guest Customer	Any 09XXXXXXXX	N/A	/	Book without registration (phone at booking = identity)

2.2 Staff Sub-Role Accounts

Staff accounts are created by Company Admins within the Selam Bus company. Log in with Company Admin credentials to access the staff management page and view/create staff accounts with these sub-roles:

Staff Role	Portal	Capabilities to Test
DRIVER	/driver/track	GPS tracking, trip departure, odometer entry
CONDUCTOR	/company/*	Ticket verification (QR scan), boarding checklist
MANUAL_TICKETER	/cashier/*	Manual ticket sales, replacement ticket sales
MECHANIC	/mechanic/*	Work orders, vehicle inspections
FINANCE	/finance/*	Financial reports, cost tracking

2.3 Telegram Bot Account

Bot	Username	How to Test
i-Ticket Bus Bot	@i_ticket_busBot	Open https://t.me/i_ticket_busBot , send /start, follow booking flow

2.4 External System Test Credentials

System	Test Mode	Notes
TeleBirr	DEMO_MODE=true on staging	Demo mode simulates payment success; production uses real TeleBirr API
SMS Gateway	N/A	Real SMS delivery; test with valid Ethiopian phone numbers
Cron Jobs	Bearer token: CRON_SECRET env var	Test via <code>GET /api/cron/cleanup</code> with <code>Authorization: Bearer {CRON_SECRET}</code>

3. API Endpoints in Scope

3.1 Public Endpoints (No Authentication)

#	Method	Endpoint	Purpose	Rate Limit
1	GET	/api/trips	Search available trips	Nginx 100r/m
2	GET	/api/trips/[id]	Get trip details	Nginx 100r/m
3	POST	/api/bookings	Create booking (guest allowed)	10/min/IP
4	GET	/api/track/[code]	Public ticket/booking tracking	Nginx 100r/m
5	GET	/api/cities	List all cities	Nginx 100r/m
6	GET	/api/cities/coordinates	City lat/lon for maps	Nginx 100r/m
7	POST	/api/auth/register	User registration	3/hour/IP
8	POST	/api/auth/[...nextauth]	Login (NextAuth)	5/30min/phone, 10/15min/IP
9	POST	/api/auth/forgot-password	Password reset request	3/hour/IP

10	POST	/api/auth/reset-password	Password reset with token	Nginx 100r/m
11	POST	/api/support/tickets	Create support ticket	Nginx 100r/m
12	GET	/api/tracking/[tripId]	Passenger bus tracking (DEPARTED trips)	Nginx 100r/m
13	POST	/api/csp-report	CSP violation reports	Nginx 100r/m

3.2 Authenticated Endpoints (JWT Session Required)

Customer Endpoints

#	Method	Endpoint	Purpose	Auth Role
14	POST	/api/payments	Initiate payment	CUSTOMER / COMPANY_ADMIN
15	GET	/api/bookings/my	List user's bookings	CUSTOMER
16	GET	/api/notifications	User notifications	Any authenticated
17	PATCH	/api/notifications/[id]	Mark notification read	Any authenticated
18	POST	/api/auth/force-change-password	Change temp password	COMPANY_ADMIN (mustChangePassword)

Company Admin Endpoints

#	Method	Endpoint	Purpose
19	GET/POST	/api/company/trips	List / create trips
20	GET/PUT/DELETE	/api/company/trips/[id]	Get / update / delete trip
21	PATCH	/api/company/trips/[id]/status	Change trip status
22	POST	/api/company/trips/[id]/depart	Mark trip DEPARTED
23	POST	/api/company/trips/[id]/complete	Mark trip COMPLETED
24	GET	/api/company/trips/[id]/boarding-status	Get boarding checklist
25	POST	/api/company/trips/[id]/no-show	Mark passengers NO_SHOW
26	POST	/api/company/trips/[id]/replacement-ticket	Sell replacement ticket
27	GET/POST	/api/company/trip-templates	Trip template CRUD
28	GET/POST	/api/company/staff	Staff management
29	GET/PUT	/api/company/staff/[id]	Staff detail/update
30	GET/POST	/api/company/vehicles	Vehicle management
31	GET/PUT	/api/company/vehicles/[id]	Vehicle detail/update
32	GET/POST	/api/company/vehicles/[id]/inspections	Vehicle inspections
33	GET/POST	/api/company/vehicles/[id]/fuel	Fuel entries
34	GET/POST	/api/company/work-orders	Work order management
35	GET/PUT	/api/company/work-orders/[id]	Work order detail/update
36	GET	/api/company/analytics/fleet-health	Fleet health dashboard
37	GET	/api/company/analytics/risk-trends	Risk trend charts
38	GET	/api/company/analytics/failure-timeline	Failure predictions
39	GET	/api/company/analytics/cost-forecast	Cost forecasting
40	GET	/api/company/analytics/vehicle-comparison	Vehicle comparisons
41	GET	/api/company/analytics/maintenance-windows	Maintenance scheduling

42	GET	/api/company/analytics/route-wear	Route wear analysis
43	GET	/api/company/analytics/compliance-calendar	Compliance tracking
44	GET	/api/company/reports/maintenance	Maintenance cost report
45	GET	/api/company/reports/maintenance/export	Export to Excel
46	GET	/api/company/reports/vehicle-tco	Total cost of ownership
47	GET	/api/company/reports/downtime	Downtime report
48	GET	/api/company/reports/fleet-analytics/export	Fleet analytics export
49	GET	/api/company/bookings	Company bookings list
50	GET	/api/company/manifests	Manifest downloads
51	GET	/api/company/messages	Company-platform chat
52	POST	/api/company/messages	Send message to platform
53	GET	/api/company/audit-logs	Company audit log
54	GET	/api/company/notifications	Company notifications

GPS Tracking Endpoints

#	Method	Endpoint	Purpose	Auth
55	POST	/api/tracking/update	Driver GPS position	Session (DRIVER/CONDUCTOR)
56	GET	/api/tracking/osmand	OsmAnd background GPS	Token (trackingToken)
57	GET	/api/tracking/fleet	Fleet map (all buses)	Session (COMPANY_ADMIN)
58	POST	/api/tracking/generate-token	OsmAnd token generation	Session (DRIVER)
59	GET	/api/tracking/active-trip	Driver's active trip	Session (DRIVER)

Ticket Verification

#	Method	Endpoint	Purpose	Auth
60	PATCH	/api/tickets/verify	Verify + mark ticket used	Session (COMPANY_ADMIN / SUPER_ADMIN)

Super Admin Endpoints

#	Method	Endpoint	Purpose
61	GET	/api/admin/stats	Platform dashboard stats
62	GET/POST	/api/admin/companies	Company management
63	GET/PUT	/api/admin/companies/[id]	Company detail/update
64	POST	/api/admin/companies/[id]/setup-staff	Create company admin
65	GET	/api/admin/trips	All trips (cross-company)
66	GET	/api/admin/bookings	All bookings with filters
67	GET	/api/admin/manifests	All manifests
68	GET	/api/admin/audit-logs	Full audit trail
69	GET/POST	/api/admin/platform-staff	Platform staff CRUD
70	GET/POST	/api/admin/sales-persons	Sales person management
71	GET	/api/admin/tax-reports	Revenue/tax analytics



72	GET	/api/admin/analytics/revenue	Revenue analytics
73	GET	/api/admin/analytics/top-companies	Top companies
74	GET	/api/admin/analytics/top-routes	Top routes
75	GET	/api/admin/analytics/monthly-trends	Monthly trends
76	GET	/api/admin/messages	All company messages
77	POST	/api/admin/messages	Send message to company

3.3 Webhook / Callback Endpoints

#	Method	Endpoint	Auth Method	Source
78	POST	/api/payments/telebirr/callback	HMAC-SHA256 signature	TeleBirr
79	POST	/api/telegram/webhook	Telegram-signed	Telegram API
80	POST	/api/sms/incoming	HMAC-SHA256 (SMS_WEBHOOK_SECRET)	SMS Gateway

3.4 Cron / Scheduled Endpoints

#	Method	Endpoint	Auth	Schedule
81	GET	/api/cron/cleanup	Bearer (CRON_SECRET)	Every 15 min
82	GET	/api/cron/trip-reminders	Bearer (CRON_SECRET)	Scheduled
83	GET	/api/cron/predictive-maintenance	Bearer (CRON_SECRET)	Daily
84	GET	/api/cron/telegram-cleanup	Bearer (CRON_SECRET)	Scheduled

4. Frontend Pages in Scope

4.1 Public Pages

#	URL	Description
1	/	Homepage (trip search, hero, features)
2	/login	Login page
3	/register	Registration page
4	/search	Trip search results
5	/book/[tripId]	Booking flow (seats, passengers, payment)
6	/tickets	Ticket view (after booking)
7	/track/[code]	Public ticket/bus tracking
8	/contact	Contact page
9	/about	About page
10	/faq	FAQ page
11	/privacy	Privacy policy
12	/terms	Terms of service
13	/verify/[shortCode]	Ticket verification page

4.2 Authenticated Pages

#	URL	Role Required	Description
14	/admin/dashboard	SUPER_ADMIN	Platform dashboard
15	/admin/companies	SUPER_ADMIN	Company management
16	/admin/trips	SUPER_ADMIN	Trip oversight
17	/admin/bookings	SUPER_ADMIN	Booking management
18	/admin/manifests	SUPER_ADMIN	Manifest management
19	/admin/audit-logs	SUPER_ADMIN	Audit trail
20	/admin/platform-staff	SUPER_ADMIN	Platform staff
21	/admin/sales	SUPER_ADMIN	Sales management
22	/admin/analytics	SUPER_ADMIN	Analytics dashboards
23	/admin/messages	SUPER_ADMIN	Company messaging
24	/company/dashboard	COMPANY_ADMIN	Company dashboard
25	/company/trips	COMPANY_ADMIN	Trip management
26	/company/trips/[id]	COMPANY_ADMIN	Trip detail (boarding, no-show)
27	/company/trips/new	COMPANY_ADMIN	Create new trip
28	/company/staff	COMPANY_ADMIN	Staff management
29	/company/vehicles	COMPANY_ADMIN	Vehicle management
30	/company/fleet-analytics	COMPANY_ADMIN	Fleet analytics dashboard
31	/company/fleet-tracking	COMPANY_ADMIN	Fleet map (live GPS)
32	/company/reports	COMPANY_ADMIN	Reports (maintenance, TCO, downtime)
33	/company/work-orders	COMPANY_ADMIN	Work order management
34	/company/settings	COMPANY_ADMIN	Company settings
35	/company/messages	COMPANY_ADMIN	Platform messaging
36	/driver/track	DRIVER	GPS tracking page
37	/cashier/dashboard	MANUAL_TICKETER	Manual ticketing
38	/mechanic/dashboard	MECHANIC	Inspections & work orders
39	/finance/dashboard	FINANCE	Financial reports
40	/sales/dashboard	SALES_PERSON	Referral dashboard

5. Testing Boundaries

5.1 In Scope

- All web application pages and API endpoints listed above
- Authentication and session management
- Authorization and access control (RBAC, company segregation)
- Input validation and injection testing
- Payment flow security (TeleBirr callback verification)
- GPS tracking data handling

- Telegram bot security
- SMS webhook security
- File upload/download (manifests, receipts, profile pictures)
- CSP and security header effectiveness
- Rate limiting effectiveness
- Service worker behavior
- PWA manifest and caching

5.2 Out of Scope

Item	Reason
AWS EC2 infrastructure (OS, kernel)	Managed by AWS; separate infrastructure audit
Cloudflare configuration	Managed CDN service; separate Cloudflare audit
PostgreSQL server internals	Database engine security; separate database audit
TeleBirr payment gateway internals	Third-party service; separate TeleBirr audit
SMS Gateway internals	Third-party service
Telegram Bot API internals	Third-party service
Physical security of server	AWS managed data center
Social engineering	Not in scope for web application testing
DDoS testing	May impact production availability

5.3 Testing Environment

Environment	URL	Notes
Production	https://i-ticket.et	Live system with real data; test carefully
Origin IP	http://54.147.33.168	Direct server access (bypasses Cloudflare)

Note: There is no separate staging environment. Testing should be performed on production with test accounts. Avoid creating excessive test data or triggering real payments (use DEMO_MODE test accounts).

6. Contact Information for Testing Coordination

Name	Role	Email	Phone
Abel Assefa	Platform Admin / CEO	[To be provided]	[To be provided]
[Technical Lead]	Developer Lead	[To be provided]	[To be provided]

Note: Please fill in the actual contact details before submission to INSA.

End of Document 4.2.3 - Testing Scope Definition



DOCUMENT

4.2.4

Security Functionality Document

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Security Functionality Document

Document: 4.2.4 - Security Functionality Document Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. User Roles and Access Control (RBAC)

1.1 Role Hierarchy

```

SUPER_ADMIN (Platform Owner)
|
+-- PlatformStaff (i-Ticket employees, JSON-based permissions)
|
+-- COMPANY_ADMIN (Bus Company)
    |
    +-- ADMIN (Company-level admin)
    +-- DRIVER
    +-- CONDUCTOR
    +-- MANUAL_TICKETER (Cashier)
    +-- MECHANIC
    +-- FINANCE

CUSTOMER (Registered user)

GUEST (No account - phone as identity)

SALES_PERSON (Separate auth table)
  
```

1.2 Role Permissions Matrix

Resource / Action	Guest	Customer	Driver	Conductor	Cashier	Mechanic	Finance	Co. Admin	Super Admin
Search trips	Yes	Yes	-	-	-	-	-	Yes	Yes
Book tickets	Yes	Yes	-	-	-	-	-	-	-
View own tickets	-	Yes	-	-	-	-	-	-	-
Track bus (public)	Yes	Yes	-	-	-	-	-	-	-
GPS tracking (driver)	-	-	Yes	Yes	-	-	-	-	-
Verify tickets (QR)	-	-	-	Yes	Yes	-	-	Yes	Yes
Manual ticket sales	-	-	-	-	Yes	-	-	Yes	-
Replacement ticket sales	-	-	-	-	Yes	-	-	Yes	-
Mark no-show	-	-	-	Yes	Yes	-	-	Yes	-
Manage trips (CRUD)	-	-	-	-	-	-	-	Yes	Yes
Manage staff	-	-	-	-	-	-	-	Yes	-
Manage vehicles	-	-	-	-	-	-	-	Yes	-
Vehicle inspections	-	-	Yes	-	-	Yes	-	Yes	-
Work orders	-	-	-	-	-	Yes	-	Yes	-
Fleet analytics	-	-	-	-	-	-	-	Yes	-
Fleet tracking (map)	-	-	-	-	-	-	-	Yes	-
Financial reports	-	-	-	-	-	-	Yes	Yes	-

View audit logs	-	-	-	-	-	-	-	Yes*	Yes
Manage companies	-	-	-	-	-	-	-	-	Yes
Platform analytics	-	-	-	-	-	-	-	-	Yes
Manage platform staff	-	-	-	-	-	-	-	-	Yes
Tax reports	-	-	-	-	-	-	-	-	Yes

*Company Admin sees only their company's audit logs (company-scoped).

1.3 Company Segregation (Data Isolation)

Rule: Every company API endpoint filters data by `companyId` extracted from the authenticated user's JWT session. No company can access another company's data.

Implementation:

```
// Every company API route follows this pattern:
const session = await getServerSession(authOptions)
const companyId = session.user.companyId // From JWT token

// All queries include companyId filter
const trips = await prisma.trip.findMany({
  where: { companyId } // Cannot access other companies
})
```

Shared resource exception: The `City` table is shared across all companies (read-only reference data).

1.4 Platform Staff Permissions (Fine-Grained)

Platform staff (i-Ticket employees) have JSON-based permissions stored in `PlatformStaff.permissions`:

```
{
  "canViewFinance": true,
  "canManageStaff": false,
  "canAccessReports": true,
  "canManageCompanies": false,
  "canViewAuditLogs": true
}
```

These are checked at the API level before granting access to specific Super Admin features.

2. Input Validation and Sanitization

2.1 Validation Framework

Library: Zod v3.25.x (runtime TypeScript schema validation)

Policy: All API inputs are validated server-side before processing. Client-side validation exists for UX but is never trusted.

2.2 Validation Examples by Endpoint

Endpoint	Field	Validation Rule
Registration	phone	Regex: <code>/^09\d{8}\$/</code> (Ethiopian format, exactly 10 digits)
Registration	password	Minimum length enforced
Registration	name	String, required



Booking	tripId	CUID format string
Booking	passengers	Array, max 5 items
Booking	passengers[].name	String, required
Booking	passengers[].nationalId	String, required
Booking	passengers[].seatNumber	Integer, positive
Payment	bookingId	CUID format string
Payment	method	Enum: "TELEBIRR" or "DEMO"
GPS Tracking	latitude	Float, -90 to 90
GPS Tracking	longitude	Float, -180 to 180
GPS Tracking	speed	Float, >= 0
Trip CRUD	price	Float, positive
Trip CRUD	totalSlots	Integer, positive
Trip CRUD	departureTime	DateTime, ISO format
Search params	Various	.nullish() (handles null from searchParams.get())
Integer params	Various	parseInt with scientific notation rejection

2.3 SQL Injection Prevention

Method: Prisma ORM exclusively. No raw SQL queries anywhere in the codebase.

Prisma automatically parameterizes all queries:

```
// This is how all DB queries are made (parameterized by Prisma)
const trip = await prisma.trip.findUnique({
  where: { id: tripId } // Prisma parameterizes this
})
```

2.4 XSS Prevention

Layer	Method
React	Auto-escapes all rendered content by default
CSP	default-src 'self'; script-src 'self' 'unsafe-inline' (unsafe-inline required by Next.js 14)
CSP	object-src 'none' (blocks plugins)
CSP	base-uri 'self' (prevents base tag injection)
Headers	X-Content-Type-Options: nosniff (prevents MIME type sniffing)
Headers	X-XSS-Protection: 1; mode=block (legacy browser protection)

2.5 CSRF Prevention

Method	Implementation
SameSite cookies	sameSite: 'lax' on NextAuth session cookie
CSP form-action	form-action 'self' restricts form submission targets
API design	All mutations use POST/PUT/PATCH/DELETE (not GET)
No CSRF tokens	Relied upon SameSite cookie protection (standard for SPA/JWT architectures)

2.6 File Upload Handling

Control	Implementation
Max body size	Nginx: <code>client_max_body_size 5M</code>
Allowed types	Profile pictures, receipt photos (validated by file extension)
Storage	Local filesystem (<code>/public/uploads/</code>)
Path traversal	Filenames sanitized, stored with generated IDs

3. Session Management

3.1 Session Configuration

Property	Value	File
Strategy	JWT (stateless)	<code>src/lib/auth.ts</code>
Provider	NextAuth.js CredentialsProvider	<code>src/lib/auth.ts</code>
Max Age	30 days (2,592,000 seconds)	<code>src/lib/auth.ts</code>
Cookie Name	<code>next-auth.session-token</code>	NextAuth default
httpOnly	<code>true</code> (not accessible via JavaScript)	<code>src/lib/auth.ts</code>
secure	<code>true</code> in production (HTTPS only)	<code>src/lib/auth.ts</code>
sameSite	<code>lax</code> (sent on navigation, not cross-origin POST)	<code>src/lib/auth.ts</code>
path	<code>/</code> (all routes)	<code>src/lib/auth.ts</code>
Domain	Not set (defaults to i-ticket.et)	<code>src/lib/auth.ts</code>

3.2 JWT Token Contents

Field	Description	Sensitivity
<code>id</code>	User ID (CUID)	Low
<code>role</code>	CUSTOMER / COMPANY_ADMIN / SUPER_ADMIN	Access control
<code>phone</code>	User phone number	PII
<code>companyId</code>	Company ID (null for customers)	Data segregation
<code>companyName</code>	Company name	Low
<code>companyLogo</code>	Company logo URL	Low
<code>staffRole</code>	Staff sub-role (DRIVER, MECHANIC, etc.)	Access control
<code>profilePicture</code>	Profile picture URL	Low
<code>nationalId</code>	National ID	PII
<code>mustChangePassword</code>	Temp password flag	Auth control
<code>platformStaffRole</code>	Platform staff role (Super Admin only)	Access control
<code>platformStaffPermissions</code>	JSON permissions (Super Admin only)	Access control

3.3 Session Lifecycle

1. LOGIN
Phone + Password -> bcrypt.compare() -> JWT token issued
Cookie: httpOnly, secure, sameSite=lax, maxAge=30 days
Rate limit counter cleared on success
2. EVERY REQUEST
Cookie sent automatically (same-origin)
JWT decoded and verified (NEXTAUTH_SECRET signature)
Session data available in API routes via getServerSession()
3. TOKEN REFRESH
On trigger="update": re-fetches user data from DB
Updates profile picture, nationalId, company info, permissions
4. LOGOUT
Cookie cleared (set to empty, maxAge=0)
No server-side session invalidation (JWT is stateless)
5. EXPIRY
After 30 days: cookie expires, user must re-login

3.4 Other Session Types

Session Type	Storage	TTL	Purpose
SMS Session	PostgreSQL (SmsSession table)	15 minutes	SMS booking conversation state
Telegram Session	PostgreSQL (TelegramSession table)	30 minutes	Telegram bot booking state
Password Reset Token	PostgreSQL (PasswordReset table)	1 hour	One-time password reset
OsmAnd Tracking Token	PostgreSQL (Trip.trackingToken)	Trip lifetime	Background GPS auth

4. Authentication Mechanisms

4.1 Password Authentication

Property	Value
Hash algorithm	bcrypt
Salt rounds	12
Library	bcryptjs
Storage	User.password (nullable for guests)
Comparison	bcryptjs.compare() (timing-safe)

4.2 Login Rate Limiting

Limiter	Threshold	Window	Lockout
Per phone	5 attempts	30 minutes	15-minute lockout
Per IP	10 attempts	15 minutes	Rejection until window expires
Cleanup	Automatic	Every 60 seconds	Expired entries removed

Implementation: In-memory Map (Node.js process). Cleared on successful login. Rate limit entries auto-cleaned hourly.

4.3 Registration Security

Control	Implementation
Rate limit	3 registrations per hour per IP
Duplicate prevention	Checks both User and SalesPerson tables for existing phone
Phone validation	Regex <code>/^09\d{8}\$/</code>
Password hashing	bcrypt (12 rounds) before storage
Referral attribution	Atomic transaction (registration + referral in one TX)

4.4 Password Reset Flow

1. User submits phone number
2. Server generates `crypto.randomBytes(32)` token
3. Token hashed with bcrypt before storage in `PasswordReset` table
4. Plain token sent via SMS to user's phone
5. User submits token + new password
6. Server compares token hash with `bcrypt.compare()`
7. If valid: update `User.password`, mark token `isUsed=true`
8. Token expires after 1 hour, single-use only
9. User existence is never revealed (same response for found/not-found)

4.5 Force Password Change

Company admins created with temporary passwords have `mustChangePassword=true`. They are redirected to the force-change-password page on every login attempt until they set a new password.

4.6 Secret Validation

At module load time (`src/lib/auth.ts`):

- `NEXTAUTH_SECRET` must be ≥ 32 characters
- Rejects placeholder values containing `your-super-secret-key` or `change-in-production`
- Application will not start with an insecure secret in production

5. Error Handling and Logging

5.1 Error Handling Strategy

Environment	Behavior
Production	Errors logged to PM2; generic error messages returned to client; no stack traces in API responses
Development	Queries, warnings, and errors logged to console; detailed error messages in responses



5.2 API Error Response Format

```
// Standard error response (production)
{
  "error": "Descriptive message without internal details"
}

// HTTP status codes used:
// 400 - Bad Request (validation failure)
// 401 - Unauthorized (no session)
// 403 - Forbidden (insufficient role/permission)
// 404 - Not Found
// 409 - Conflict (duplicate, concurrent modification)
// 429 - Too Many Requests (rate limited)
// 500 - Internal Server Error (generic, no details)
```

5.3 Rate Limit Error Response

```
HTTP 429 Too Many Requests
Headers:
  Retry-After: <seconds until rate limit resets>
Body:
  { "error": "Too many requests. Please try again later." }
```

5.4 Audit Logging (AdminLog)

All sensitive operations are logged to the `AdminLog` table:

Action Category	Logged Events
Authentication	Login attempts (success/failure with IP), registration, password resets
Booking	Booking creation, payment success/failure, cancellation
Trip Management	Trip creation, status changes (DEPARTED, COMPLETED, CANCELLED), auto-halt triggers
No-Show	Passenger marked NO_SHOW, replacement ticket sold
Staff Management	Staff created, role changed, status changed
Company Management	Company activated/deactivated, admin created
Financial	Commission approved, payout processed
System	Cron job status transitions, auto-halt triggers, auto-resume

Log entry structure:

```
{
  "id": "cuid",
  "userId": "who performed the action",
  "action": "ACTION_TYPE",
  "details": "Human-readable description with context",
  "tripId": "related trip (optional)",
  "companyId": "related company (optional)",
  "createdAt": "timestamp"
}
```

5.5 CSP Violation Logging

CSP violations are reported to `/api/csp-report` and logged via `console.warn('[CSP Violation]', ...)` to PM2 output logs.

5.6 Application Logs

Log Type	Location	Retention
PM2 Error Log	/var/log/pm2/i-ticket-error.log	Until log rotation
PM2 Output Log	/var/log/pm2/i-ticket-out.log	Until log rotation
Nginx Access Log	/var/log/nginx/access.log	System logrotate
Nginx Error Log	/var/log/nginx/error.log	System logrotate
AdminLog (DB)	PostgreSQL AdminLog table	Indefinite
ProcessedCallback (DB)	PostgreSQL ProcessedCallback table	Indefinite

6. Secure Communications (TLS/SSL)

6.1 TLS Configuration

Property	Value
SSL Mode	Cloudflare Full (Strict)
Minimum TLS	1.2 (enforced at Cloudflare edge)
Certificate	Cloudflare Universal SSL (edge) + Cloudflare Origin Certificate (server)
HSTS	Enabled: max-age=31536000; includeSubDomains; preload
HSTS (App Level)	Strict-Transport-Security: max-age=63072000; includeSubDomains; preload (2 years)
Upgrade	CSP upgrade-insecure-requests directive

6.2 Traffic Flow

```

Client Browser
  |
  | TLS 1.2+ (Cloudflare terminates)
  |
Cloudflare Edge (SSL Full Strict)
  |
  | HTTPS (Cloudflare Origin Certificate)
  |
Nginx (Port 80 internally, SSL handled by Cloudflare)
  |
  | HTTP localhost:3000
  |
Node.js / Next.js
  |
  | Prisma ORM (localhost PostgreSQL, no TLS needed for local)
  |
PostgreSQL 16.11 (localhost:5432)
  
```

6.3 External Service Communication

Service	Protocol	Certificate Validation
TeleBirr API	HTTPS	Node.js default CA bundle
SMS Gateway	HTTPS	Node.js default CA bundle
Telegram API	HTTPS	Node.js default CA bundle
ClickUp API	HTTPS	Node.js default CA bundle

OSRM API	HTTPS	Node.js default CA bundle
Nominatim API	HTTPS	Node.js default CA bundle
OpenStreetMap Tiles	HTTPS	Browser CA bundle

6.4 Security Headers (Complete)

Header	Value	Purpose
Strict-Transport-Security	max-age=63072000; includeSubDomains; preload	Force HTTPS for 2 years
X-Frame-Options	DENY	Prevent clickjacking
X-Content-Type-Options	nosniff	Prevent MIME sniffing
X-XSS-Protection	1; mode=block	Legacy XSS filter
Referrer-Policy	strict-origin-when-cross-origin	Limit referrer leakage
Permissions-Policy	camera=(), microphone=(), geolocation=(self), payment=(), usb=(), bluetooth=(), magnetometer=(), gyroscope=(), accelerometer=()	Restrict browser APIs
Content-Security-Policy	(see Section 2.4 above for full policy)	Restrict content sources
Cache-Control	no-store (all pages); no-store, no-cache, must-revalidate, proxy-revalidate (auth pages)	Prevent caching of sensitive content
Pragma	no-cache	HTTP/1.0 cache prevention
Expires	0 (auth pages only)	Force revalidation
X-Powered-By	REMOVED	Hide server technology
Server	STRIPPED by Nginx	Hide server software

7. Payment Security Controls

7.1 TeleBirr Callback Verification (5 Security Gates)

```

Gate 1: CALLBACK HASH
SHA-256(JSON.stringify(sortedPayload))
Purpose: Canonical representation for deduplication

Gate 2: REPLAY DETECTION
Query ProcessedCallback by transactionId
If found: return 200 (idempotent, no reprocessing)
Uses unique indexes on transactionId AND callbackHash

Gate 3: SIGNATURE VERIFICATION
Compute: HMAC-SHA256(payload, TELEBIRR_APP_KEY)
Compare: crypto.timingSafeEqual(computed, received)
Purpose: Verify callback authenticity
Note: timing-safe comparison prevents timing attacks

Gate 4: TIMESTAMP VALIDATION
|current_time - callback_timestamp| < 5 minutes
Purpose: Prevent stale callback replay
Note: logs warning but still processes (clock skew tolerance)

Gate 5: IP WHITELIST (Optional)
If TELEBIRR_WEBHOOK_IPS env is set:
Verify request IP is in the allowed list
Purpose: Additional layer of origin verification

```

7.2 Payment Flow Security

Control	Implementation
Payment window	10-minute timeout (enforced server-side)
Amount verification	Server recalculates: ticket price + 5% commission + 15% VAT
Rate limiting	3 payment attempts per hour per booking
Enhanced rate limiting	Per-IP + per-user + per-booking limits
Ownership check	Booking must belong to requesting user
Concurrent protection	Row-level locking on booking during payment
Demo mode guard	<code>DEMO_MODE</code> blocked when <code>NODE_ENV=production</code>
Nonce	<code>crypto.randomBytes(16)</code> per payment request
Idempotency	ProcessedCallback table records every callback (inside transaction)

7.3 Commission Security

Aspect	Implementation
Calculation	Always server-side (never from client input)
Formula	Base: 5% of ticket price; VAT: 15% of commission; TeleBirr fee: 0.5% (absorbed by platform)
Replacement sales	Commission = 0 (matches cashier pattern)
Sales commission	5% of platform's 5% = 0.25% of ticket; 70/30 recruiter split
Audit trail	SalesCommission table with status tracking (PENDING -> APPROVED -> PAID)



8. Concurrency and Data Integrity Controls

8.1 Row-Level Locking

```
-- Used when creating bookings (prevents double-selling seats)
SELECT * FROM "Trip" WHERE id = $1 FOR UPDATE NOWAIT
```

`NOWAIT` : If another transaction holds the lock, immediately fails (no waiting/deadlock). The application catches this and returns a conflict error to the user.

8.2 Transaction Timeouts

```
// Dual timeout pattern:
const result = await Promise.race([
  prisma.$transaction(callback, {
    maxWait: 5000, // 5s to acquire transaction
    timeout: 10000 // 10s to complete transaction
  }),
  new Promise((_, reject) =>
    setTimeout(() => reject(new Error('Transaction timeout')), 10000)
  )
])
```

8.3 Optimistic Locking

`Trip.version` field incremented on every update. Concurrent modifications detected and rejected.

8.4 Idempotency

Operation	Idempotency Mechanism
Payment callback	ProcessedCallback.transactionId (unique index)
Payment callback	ProcessedCallback.callbackHash (unique index)
No-show marking	Re-marking already NO_SHOW passenger is a no-op
Ticket verification	Checks <code>isUsed</code> flag before marking

9. Data Protection

9.1 Sensitive Data Handling

Data Type	At Rest	In Transit	Access Control
Passwords	bcrypt hash (12 rounds)	HTTPS/TLS 1.2+	Never exposed in API responses
National IDs	Plaintext in DB	HTTPS/TLS 1.2+	Company admin + super admin only
Phone numbers	Plaintext in DB	HTTPS/TLS 1.2+	Varies by context
Bank accounts	Plaintext in DB	HTTPS/TLS 1.2+	Company admin + super admin only
Payment data	Amount + method in DB	HTTPS/TLS 1.2+	Booking owner + admin
GPS positions	Lat/lon in DB	HTTPS/TLS 1.2+	Public for DEPARTED trips; auto-purged after 7 days
Reset tokens	bcrypt hash in DB	HTTPS/TLS 1.2+ (via SMS)	Single-use, 1-hour expiry
QR codes	Base64 data URL in DB	HTTPS/TLS 1.2+	Ticket owner + verifying staff

Ticket shortCodes	Plaintext in DB	HTTPS/TLS 1.2+	Public (verification URL)
Session cookies	JWT (signed, not encrypted)	HTTPS only (secure flag)	httpOnly (no JS access)

9.2 Data Retention

Data	Retention	Cleanup Mechanism
GPS positions (TripPosition)	7 days after trip ends	Cron job auto-purge
SMS sessions	15 minutes	Cron job cleanup
Telegram sessions	30 minutes	Cron job cleanup
Password reset tokens	1 hour	Expiry + isUsed flag
Pending payments	10 minutes	Cron job cancellation + seat release
Pending bookings	10 minutes	Cron job cancellation + seat release
Audit logs (AdminLog)	Indefinite	No automatic cleanup
Payment callbacks (ProcessedCallback)	Indefinite	No automatic cleanup

10. Technical Function Descriptions

10.1 Core Security Functions

Function	File	Description
<code>authOptions</code>	<code>src/lib/auth.ts</code>	NextAuth configuration: credentials provider, JWT strategy, 30-day sessions, rate limiting, secret validation
<code>transactionWithTimeout()</code>	<code>src/lib/db.ts</code>	Prisma transaction wrapper with dual timeout (Promise.race + native), 10s default
<code>checkRateLimit()</code>	<code>src/lib/rate-limit.ts</code>	In-memory rate limiter with per-endpoint configs, emergency cleanup at 80% capacity
<code>checkEnhancedRateLimit()</code>	<code>src/lib/rate-limit.ts</code>	Dual IP + user rate limiting for sensitive operations
<code>checkBookingRateLimit()</code>	<code>src/lib/rate-limit.ts</code>	Per-booking payment rate limiting
<code>signTelebirrRequest()</code>	<code>src/lib/payments/telebirr.ts</code>	HMAC-SHA256 signing for outbound TeleBirr requests
<code>verifyTelebirrSignature()</code>	<code>src/lib/payments/telebirr.ts</code>	HMAC-SHA256 verification with <code>crypto.timingSafeEqual()</code>
<code>computeCallbackHash()</code>	<code>src/lib/payments/callback-hash.ts</code>	SHA-256 hash of payment callback payload for deduplication
<code>hasDepartedEthiopia()</code>	<code>src/lib/utils.ts</code>	Timezone-safe departure check (prevents UTC/Ethiopia mismatch)
<code>isTodayEthiopia()</code>	<code>src/lib/utils.ts</code>	Timezone-safe date comparison for Ethiopia (UTC+3)

10.2 CSP Middleware Function

Property	Value
File	<code>src/middleware.ts</code>
Scope	All page routes (excludes <code>/api</code> , <code>/_next/static</code> , <code>favicon.ico</code>)
Function	Injects CSP header, Cache-Control, Report-To, Reporting-Endpoints on every page response

10.3 Cron Security Functions

Function	File	Auth	Description
----------	------	------	-------------

Cleanup	/api/cron/cleanup	Bearer CRON_SECRET	Timeout payments, cancel stale bookings, release seats, auto-transition trip statuses, reset staff, purge GPS data
Trip Reminders	/api/cron/trip-reminders	Bearer CRON_SECRET	Send departure reminders via SMS
Predictive Maintenance	/api/cron/predictive-maintenance	Bearer CRON_SECRET	AI risk score recalculation for all vehicles
Telegram Cleanup	/api/cron/telegram-cleanup	Bearer CRON_SECRET	Delete expired Telegram sessions

End of Document 4.2.4 - Security Functionality Document





DOCUMENT

4.2.5

Secure Coding Standards

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Secure Coding Standard Documentation

Document: 4.2.5 - Secure Coding Standard Documentation Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Secure Coding Guidelines

i-Ticket development follows OWASP Secure Coding Practices aligned with the OWASP Top 10 (2021). The following guidelines are enforced across the codebase.

1.1 Guiding Principles

Principle	Implementation
Defense in Depth	9 security layers: Cloudflare -> Nginx -> Middleware -> Headers -> Auth -> Validation -> Payment Security -> DB Security -> Audit
Least Privilege	RBAC with company segregation; staff sub-roles limit access to needed functions only
Fail Securely	All errors return generic messages; transactions roll back on failure; seats released on payment timeout
Don't Trust Client	Server-side validation on all inputs; server-side amount recalculation; server-side role checks
Separation of Concerns	Business logic in <code>src/lib/</code> , validation at API boundary, auth in middleware

2. Internal Rules and Checklists

2.1 Mandatory Rules (from RULES.md)

Rule ID	Rule	Enforcement
RULE-001	Company Segregation: Every company API MUST filter by <code>companyId</code>	Code review; session.user.companyId used in every query
RULE-002	View-Only Trips: DEPARTED/COMPLETED/CANCELLED/SOLD-OUT trips are read-only	Status guard checks in all trip mutation APIs
RULE-003	Guest Booking: Phone payment = verification, no OTP required	Business rule enforced in booking API
RULE-004	Auto-Halt: Manual ticketing never blocked; online halts at ≤ 10 slots	Conditional logic with bypass flags
RULE-005	No-Show: Only on DEPARTED trips; <code>availableSlots</code> never modified	Trip status guard + transaction-safe counters

2.2 Pre-Development Checklist

Before writing any API endpoint or data mutation:

- Read RULES.md appendices (File-to-Rule mapping)
- Check Bug Registry for known issues in the area
- Identify which user roles can access this endpoint
- Plan company segregation (`companyId` filter)
- Define Zod validation schema for all inputs
- Plan error responses (no internal details)
- Consider concurrency (need row locking? transaction?)
- Consider rate limiting requirements
- Plan AdminLog entries for audit trail
- Verify timezone handling (use Ethiopia timezone utilities)

2.3 Code Review Checklist

- All API inputs validated with Zod schemas
- No raw SQL queries (Prisma ORM only)
- Company segregation enforced (companyId in WHERE clause)
- Role/permission checks present for authenticated endpoints
- No sensitive data in error responses
- Financial amounts recalculated server-side (never from client)
- AdminLog entries for sensitive operations
- Rate limiting applied to mutation endpoints
- Timezone-safe date comparisons (Ethiopia utilities)
- No `parseInt` without scientific notation check
- No `searchParams.get()` without `.nullish()` handling
- Transaction timeouts for multi-step DB writes

3. SQL Injection Prevention

3.1 Policy

All database access **MUST** go through Prisma ORM. Raw SQL queries are prohibited.

3.2 Implementation

```
// CORRECT: Prisma parameterized query (ALL queries use this pattern)
const trip = await prisma.trip.findUnique({
  where: { id: userInput } // Prisma auto-parameterizes
})

// PROHIBITED: Raw SQL (never used in codebase)
// prisma.$queryRaw`SELECT * FROM "Trip" WHERE id = ${userInput}`
// prisma.$executeRawUnsafe(...)
```

3.3 Verification

A grep for `$queryRaw`, `$executeRaw`, `$queryRawUnsafe`, or `$executeRawUnsafe` across the entire codebase returns zero results. All database operations use Prisma's type-safe query builder.

4. XSS Prevention

4.1 Output Encoding

Layer	Method	Coverage
React 18	Auto-escapes all JSX expressions (<code>{variable}</code>)	All rendered content
React	<code>dangerouslySetInnerHTML</code> not used in codebase	N/A
CSP	<code>script-src 'self' 'unsafe-inline'</code>	Limits script sources
CSP	<code>default-src 'self'</code>	Blocks external content by default
CSP	<code>object-src 'none'</code>	Blocks Flash/Java plugins

4.2 Input Sanitization

Input Type	Handling
Text fields (name, message)	Zod string validation; React auto-escaping on render
Phone numbers	Regex <code>/^\d{10}/</code> (only digits)
Integer IDs	<code>parseInt</code> with scientific notation rejection
Search params	Zod <code>.nullish()</code> (handles null from <code>searchParams.get()</code>)
JSON fields	Zod object/array validation before <code>JSON.stringify</code> storage
URLs	Not user-controlled (all URLs are system-generated)

4.3 CSP as Defense Layer

```
Content-Security-Policy:
  default-src 'self';
  script-src 'self' 'unsafe-inline' https://static.cloudflareinsights.com;
  style-src 'self' 'unsafe-inline' https://fonts.googleapis.com;
  img-src 'self' data: https://api.qrserver.com https://*.tile.openstreetmap.org https://unpkg.com;
  font-src 'self' data: https://fonts.gstatic.com;
  connect-src 'self' https://api.telebirr.com https://api.webbirr.com
    https://*.tile.openstreetmap.org https://nominatim.openstreetmap.org
    https://cloudflareinsights.com https://router.project-osrm.org;
  media-src 'self' data: blob:;
  frame-ancestors 'none';
  base-uri 'self';
  form-action 'self';
  object-src 'none';
  upgrade-insecure-requests;
```

Note on `unsafe-inline`: Required for `script-src` because Next.js 14 does not propagate nonces to inline `<script>` tags. Full nonce-based CSP requires Next.js 15+. This is a known framework limitation documented in the Bug Registry.

5. CSRF Prevention

5.1 Strategy

SameSite cookie-based protection (standard for JWT/SPA architectures):

Control	Value	Effect
<code>sameSite</code>	<code>lax</code>	Cookie sent on same-site navigations; NOT sent on cross-origin POST/PUT/DELETE
<code>httpOnly</code>	<code>true</code>	Cookie not accessible via JavaScript (prevents XSS-based theft)
CSP <code>form-action</code>	<code>'self'</code>	Forms can only submit to same origin
API design	POST/PUT/PATCH/DELETE for mutations	GET requests never mutate state

5.2 Why No CSRF Tokens

NextAuth.js with JWT strategy and `sameSite: lax` cookies provides equivalent protection to CSRF tokens:

- Cross-origin requests (from attacker sites) do not include the session cookie
- `form-action 'self'` CSP prevents form-based CSRF
- All mutations require POST/PUT/PATCH/DELETE methods

6. Secure Input Handling

6.1 Zod Validation Pattern

Every API endpoint follows this pattern:

```
// 1. Define schema
const schema = z.object({
  tripId: z.string().cuid(),
  passengers: z.array(z.object({
    name: z.string().min(1),
    nationalId: z.string().min(1),
    phone: z.string().regex(/^09\d{8}$/),
    seatNumber: z.number().int().positive().optional(),
  })).min(1).max(5),
})

// 2. Parse and validate (throws on invalid input)
const data = schema.parse(await request.json())

// 3. Use validated data (type-safe from here)
const booking = await createBooking(data)
```

6.2 Common Validation Patterns

Pattern	Usage	Why
.nullish() not .optional()	URL search params	searchParams.get() returns null, not undefined
Custom parseInt validator	Integer params	Native parseInt accepts scientific notation (1e2 = 100)
.cuid()	ID parameters	Validates CUID format, rejects arbitrary strings
.regex(/^09\d{8}\$/)	Phone numbers	Exact Ethiopian mobile format
.min(1).max(5)	Passenger arrays	Business rule: 1-5 passengers per booking
.enum([...])	Status fields	Only accept known values

6.3 Integer Validation (Scientific Notation Guard)

```
// PROBLEM: parseInt("1e2") returns 1 (truncates at 'e')
// This could bypass max-value checks

// SOLUTION: Custom validator used throughout codebase
function safeParseInt(value: string): number | null {
  if (/eE/.test(value)) return null // Reject scientific notation
  const num = parseInt(value, 10)
  if (isNaN(num)) return null
  return num
}
```

7. File Upload and Download Controls

7.1 Upload Security

Control	Implementation
Max file size	Nginx: client_max_body_size 5M

Storage location	/public/uploads/ (local filesystem)
Filename generation	System-generated IDs (no user-controlled filenames)
Path traversal	Filenames sanitized; no directory navigation allowed
Serving	Next.js static file serving from /public/

7.2 Download Security (Manifests, Reports)

Control	Implementation
Manifest files	Stored in /public/manifests/company-{companyId}/
Company segregation	File paths include companyId; access checked before serving
Excel generation	ExcelJS library generates .xlsx in memory; streamed to client
Content-Type	Proper MIME types set (application/vnd.openxmlformats-officedocument.spreadsheetml.sheet)
No user-controlled paths	File paths constructed server-side from validated IDs

8. Authentication and Session Management Standards

8.1 Password Standards

Property	Standard
Hashing	bcrypt with 12 salt rounds
Storage	Hash only (plaintext never stored)
Comparison	bcryptjs.compare() (timing-safe)
Reset tokens	Crypto-random (32 bytes), bcrypt-hashed before storage
Guest users	password: null (phone payment = verification)
Temp passwords	mustChangePassword: true flag forces change on login

8.2 Session Standards

Property	Standard
Strategy	JWT (stateless)
Cookie flags	httpOnly: true, secure: true (production), sameSite: 'lax'
Max age	30 days
Secret	>= 32 characters, validated at startup, placeholder rejection
Token contents	Minimal PII (id, role, companyId, staffRole)
Invalidation	Client-side cookie clearing (JWT has no server-side revocation)

8.3 Rate Limiting Standards

Endpoint Type	Limit	Window
Login	5 per phone / 10 per IP	30 min / 15 min
Registration	3 per IP	1 hour
Password reset	3 per IP	1 hour



Booking creation	10 per IP	1 minute
Payment	5 per IP	1 minute
Payment (per booking)	3 per booking	1 hour
Support ticket	5 per IP	1 hour
Webhooks	100 per source	1 minute
Default API	60 per IP	1 minute

9. Regular Patching and Library Validation

9.1 Dependency Management

Practice	Implementation
Package manager	npm with <code>package-lock.json</code> for deterministic installs
Production install	<code>npm ci</code> (clean install from lockfile)
Vulnerability scanning	GitHub Dependabot (automated alerts)
Known overrides	<code>glob: ^10.5.0</code> (security patch applied)
Resolution status	6 of 7 Dependabot alerts resolved as of v2.13.0

9.2 Runtime Versions

Component	Version	LTS Status
Node.js	20.20.0	Active LTS (supported until April 2026)
Next.js	14.2.x	Latest 14.x stable
PostgreSQL	16.11	Supported (EOL: November 2028)
Ubuntu	22.04 LTS	Supported (EOL: April 2027)

9.3 Update Procedure

1. Review Dependabot alerts on GitHub
2. Update dependencies locally: `npm update` / `npm audit fix`
3. Run local build: `npm run build`
4. Test critical flows (booking, payment, tracking)
5. Commit and push
6. Deploy to production (see CLAUDE.md deployment workflow)
7. Verify: `pm2 logs i-ticket --lines 20`

10. Secure Communication Patterns

10.1 API Request Signing (Outbound)

- TeleBirr Payment Initiation:
1. Construct payload (`appId`, `merchantCode`, `amount`, `phone`, `nonce`, `timestamp`)
 2. Sort parameters alphabetically
 3. Sign: HMAC-SHA256(`sortedPayload`, `TELEBIRR_APP_KEY`)
 4. Include signature in request header
 5. Send via HTTPS



10.2 Webhook Verification (Inbound)

```
TeleBirr Callback:
1. Receive POST body
2. Compute: HMAC-SHA256(payload, TELEBIRR_APP_KEY)
3. Compare: crypto.timingSafeEqual(computed, received)
4. Check replay: query ProcessedCallback table
5. Validate timestamp (5-min window)
6. Optional: IP whitelist check
```

```
SMS Inbound Webhook:
1. Receive POST body
2. Compute: HMAC-SHA256(payload, SMS_WEBHOOK_SECRET)
3. Compare with received signature header
4. Process if valid
```

10.3 Timing-Safe Comparison

All cryptographic comparisons use `crypto.timingSafeEqual()` to prevent timing side-channel attacks:

```
const isValid = crypto.timingSafeEqual(
  Buffer.from(computedSignature, 'hex'),
  Buffer.from(receivedSignature, 'hex')
)
```

11. Known Limitations and Mitigations

Limitation	Risk	Mitigation
<code>unsafe-inline</code> in CSP script-src	Allows inline scripts (XSS vector)	Next.js 14 limitation; React auto-escaping + Zod validation compensate; planned fix with Next.js 15 nonce support
JWT has no server-side revocation	Compromised token valid until expiry (30 days)	Short-lived tokens not feasible due to UX; password change does not invalidate existing tokens
In-memory rate limiter	Resets on server restart/PM2 restart	Daily restart at 3 AM provides natural reset; Nginx rate limiting provides baseline protection
No field-level encryption	Sensitive fields (nationalId, bankAccount) stored plaintext	TLS in transit; DB on localhost (no network exposure); recommended for future implementation
No SIEM integration	No centralized log analysis	AdminLog + PM2 logs provide audit trail; Cloudflare analytics for traffic anomalies
Single EC2 instance	No redundancy	PM2 auto-restart; Cloudflare caching for static assets; acceptable for current traffic volume
No database encryption at rest	Data on disk unencrypted	AWS EBS provides volume-level encryption option (not currently enabled); recommended for future

12. Bug Registry (Security-Relevant)

Known bugs with security implications that have been documented and fixed:

Bug	Root Cause	Fix	Version
Grey map tiles in PWA	Service worker intercepted cross-origin tile requests	Skip non-same-origin URLs in sw.js	v2.12.2
UTC vs Ethiopia timezone	AWS EC2 runs UTC; date comparisons 3 hours early	<code>hasDepartedEthiopia()</code> timezone utility	v2.10.15



Nonce CSP breaks Next.js 14	Nextjs 14 doesn't propagate nonces to inline scripts	Use <code>'self' 'unsafe-inline'</code> instead; nonces need Nextjs 15+	v2.10.17
Cloudflare Email Obfuscation	Modifies HTML, breaks React hydration	Disabled in Cloudflare dashboard	v2.12.1
Stale closure in polling	<code>setInterval</code> captures state at setup time	Use <code>useRef</code> for current state in intervals	v2.10.7
Cron duration unit mismatch	DB stores minutes; code treated as hours	<code>trip.estimatedDuration * 60 * 1000</code>	v2.10.14
Zod null vs undefined	<code>searchParams.get()</code> returns null	Use <code>.nullish()</code> not <code>.optional()</code>	v2.10.5
<code>parseInt</code> scientific notation	<code>parseInt("1e2")</code> returns 1	Custom validator rejects [eE]	v2.10.x

End of Document 4.2.5 - Secure Coding Standard Documentation





D O C U M E N T

5.4.1

API Request Response Samples

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION



i-Ticket Platform - API Request/Response Samples

Document: 5.4.1 - Request/Response File Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. User Registration

Endpoint: `POST /api/auth/register`

Authentication: None (public) Rate Limit: 3 requests/hour per IP

Successful Registration

Request:

```
POST /api/auth/register HTTP/1.1
Host: i-ticket.et
Content-Type: application/json

{
  "name": "Abebe Bikila",
  "phone": "0912345678",
  "email": "abebe@example.com",
  "password": "securePassword123",
  "referralCode": "ABEL23"
}
```

Response (201 Created):

```
{
  "message": "User created successfully",
  "user": {
    "id": "clx1abc2300001234def56gh",
    "name": "Abebe Bikila",
    "phone": "0912345678"
  },
  "referred": true,
  "isSalesPerson": false
}
```

Failed Registration (Duplicate Phone)

Request: Same as above with existing phone

Response (409 Conflict):

```
{
  "error": "This phone number is already registered"
}
```

Failed Registration (Invalid Phone Format)

Response (400 Bad Request):

```
{  
    "error": "Invalid phone number format"  
}
```

Failed Registration (Rate Limited)

Response (429 Too Many Requests):

```
HTTP/1.1 429 Too Many Requests  
Retry-After: 3600  
  
{  
    "error": "Too many requests. Please try again later."  
}
```

2. Trip Search

Endpoint: `GET /api/trips`

Authentication: None (public)

Successful Search

Request:

```
GET /api/trips?origin=Addis&destination=Hawassa&date=2026-02-20&page=1&limit=20&busType=STANDARD&sortBy=price HTTP/1.1  
Host: i-ticket.et
```

Response (200 OK):



```
{  
  "trips": [  
    {  
      "id": "clx1trip0100001234abc56de",  
      "origin": "Addis Ababa",  
      "destination": "Hawassa",  
      "route": "Addis Ababa - Ziway - Hawassa",  
      "intermediateStops": "Ziway,Shashemene",  
      "departureTime": "2026-02-20T06:00:00.000Z",  
      "estimatedDuration": 240,  
      "distance": 275,  
      "price": 1000,  
      "busType": "STANDARD",  
      "totalSlots": 45,  
      "availableSlots": 32,  
      "hasWater": true,  
      "hasFood": false,  
      "defaultPickup": "Meskel Square Bus Station",  
      "defaultDropoff": "Hawassa Bus Terminal",  
      "status": "SCHEDULED",  
      "company": {  
        "id": "clx1comp0100001234abc56de",  
        "name": "Selam Bus",  
        "logo": "/uploads/company-logos/selam.png"  
      }  
    }  
  ],  
  "pagination": {  
    "page": 1,  
    "limit": 20,  
    "total": 47,  
    "pages": 3  
  }  
}
```

Built-in Filters (always applied server-side):

- `isActive = true`, `bookingHalted = false`, `availableSlots > 0`
- Future trips only (or remaining today's trips in Ethiopia timezone)

3. Create Booking

Endpoint: `POST /api/bookings`

Authentication: JWT session (registered user) OR guest checkout (phone as identity) Rate Limit: 10 requests/min per IP

Successful Booking (Authenticated User)

Request:



```
POST /api/bookings HTTP/1.1
Host: i-ticket.et
Content-Type: application/json
Cookie: next-auth.session-token=eyJhbGc...
Content-Length: 333

{
  "tripId": "clx1trip010001234abc56de",
  "passengers": [
    {
      "name": "Abebe Bikila",
      "nationalId": "ET-123456",
      "phone": "0912345678",
      "seatNumber": 5,
      "pickupLocation": "Meskel Square",
      "dropoffLocation": "Hawassa Bus Terminal"
    },
    {
      "name": "Sara Bekele",
      "nationalId": "ET-654321",
      "phone": "0912345679",
      "seatNumber": 6,
      "isChild": false
    }
  ],
  "selectedSeats": [5, 6],
  "totalAmount": 2120
}
```

Response (201 Created):



```
{  
  "booking": {  
    "id": "clx1book0100001234abc56de",  
    "tripId": "clx1trip0100001234abc56de",  
    "userId": "clx1user0100001234abc56de",  
    "totalAmount": 2120,  
    "commission": 100,  
    "commissionVAT": 15,  
    "status": "PENDING",  
    "createdAt": "2026-02-19T10:00:00.000Z",  
    "updatedAt": "2026-02-19T10:00:00.000Z",  
    "passengers": [  
      {  
        "id": "clx1pass0100001234abc56de",  
        "name": "Abebe Bikila",  
        "nationalId": "ET-123456",  
        "phone": "0912345678",  
        "seatNumber": 5,  
        "specialNeeds": null,  
        "pickupLocation": "Meskel Square",  
        "dropoffLocation": "Hawassa Bus Terminal",  
        "boardingStatus": "PENDING"  
      },  
      {  
        "id": "clx1pass0200001234abc56de",  
        "name": "Sara Bekele",  
        "nationalId": "ET-654321",  
        "phone": "0912345679",  
        "seatNumber": 6,  
        "specialNeeds": null,  
        "pickupLocation": null,  
        "dropoffLocation": null,  
        "boardingStatus": "PENDING"  
      }  
    ],  
    "trip": {  
      "id": "clx1trip0100001234abc56de",  
      "origin": "Addis Ababa",  
      "destination": "Hawassa",  
      "departureTime": "2026-02-20T06:00:00.000Z",  
      "price": 1000,  
      "company": {  
        "id": "clx1comp0100001234abc56de",  
        "name": "Selam Bus"  
      }  
    }  
  }  
}
```

Note: `totalAmount` and `commission` are recalculated server-side regardless of what the client sends.

Failed Booking (Seat Conflict)

Response (409 Conflict):

```
{  
  "error": "This trip is being booked by another user. Please try again in a moment."  
}
```

Failed Booking (Trip Halted)

Response (400 Bad Request):

```
{  
  "error": "Booking is currently halted for this trip"  
}
```



4. Process Payment

Endpoint: `POST /api/payments`

Authentication: JWT session (or guest via booking ownership) Rate Limit: Enhanced (IP + user + per-booking: 3 attempts/hour)

Successful Payment (Demo Mode)

Request:

```
POST /api/payments HTTP/1.1
Host: i-ticket.et
Content-Type: application/json
Cookie: next-auth.session-token=eyJhbGc...
{
  "bookingId": "clx1book0100001234abc56de",
  "method": "DEMO"
}
```

Response (200 OK):

```
{
  "success": true,
  "payment": {
    "id": "clx1pay00100001234abc56de",
    "bookingId": "clx1book0100001234abc56de",
    "amount": 2120,
    "method": "DEMO",
    "transactionId": "TXN-170830000000-a1b2c3d4",
    "status": "SUCCESS",
    "createdAt": "2026-02-19T10:01:00.000Z"
  },
  "tickets": [
    {
      "id": "clx1tick0100001234abc56de",
      "bookingId": "clx1book0100001234abc56de",
      "tripId": "clx1trip0100001234abc56de",
      "passengerName": "Abebe Bikila",
      "seatNumber": 5,
      "qrCode": "data:image/png;base64,iVBORw0KGgo...",
      "shortCode": "ABC123",
      "isUsed": false,
      "createdAt": "2026-02-19T10:01:00.000Z"
    },
    {
      "id": "clx1tick02000001234abc56de",
      "bookingId": "clx1book0100001234abc56de",
      "tripId": "clx1trip0100001234abc56de",
      "passengerName": "Sara Bekele",
      "seatNumber": 6,
      "qrCode": "data:image/png;base64,iVBORw0KGgo...",
      "shortCode": "DEF456",
      "isUsed": false,
      "createdAt": "2026-02-19T10:01:00.000Z"
    }
  ]
}
```

Failed Payment (Expired Window)

Response (400 Bad Request):



```
{  
  "error": "Payment window expired",  
  "message": "This booking was created more than 10 minutes ago and has expired. Please create a new booking."  
}
```

Failed Payment (Already Paid)

Response (400 Bad Request):

```
{  
  "error": "Booking already paid"  
}
```

5. TeleBirr Payment Callback

Endpoint: `POST /api/payments/telebirr/callback`

Authentication: HMAC-SHA256 signature + optional IP whitelist Security: 5-gate verification (hash, replay, signature, timestamp, IP)

Successful Callback

Request (from TeleBirr servers):

```
POST /api/payments/telebirr/callback HTTP/1.1  
Host: i-ticket.et  
Content-Type: application/json  
X-Forwarded-For: 196.188.xxx.xxx  
  
{  
  "transactionId": "TB_20260219100200_123456",  
  "outTradeNo": "clx1book0100001234abc56de",  
  "status": "SUCCESS",  
  "amount": "2120.00",  
  "currency": "ETB",  
  "timestamp": "2026-02-19T10:02:00Z",  
  "signature": "a1b2c3d4e5f6..."  
}
```

Response (200 OK):

```
{  
  "success": true  
}
```

Replay (Already Processed)

Response (200 OK) — idempotent:

```
{  
  "success": true,  
  "message": "Already processed",  
  "processedAt": "2026-02-19T10:02:01.000Z"  
}
```



Invalid Signature

Response (401 Unauthorized):

```
{  
  "error": "Invalid signature"  
}
```

Unauthorized IP

Response (403 Forbidden):

```
{  
  "error": "Unauthorized IP address"  
}
```

6. Ticket Verification (QR Scan)

Endpoint: `POST /api/tickets/verify`

Authentication: JWT session (COMPANY_ADMIN or SUPER_ADMIN)

Valid Ticket

Request:

```
POST /api/tickets/verify HTTP/1.1  
Host: i-ticket.et  
Content-Type: application/json  
Cookie: next-auth.session-token=eyJhbGc...  
  
{  
  "code": "ABC123"  
}
```

Response (200 OK):



```
{  
    "valid": true,  
    "message": "Ticket is valid and ready for boarding",  
    "ticket": {  
        "id": "clx1tick010001234abc56de",  
        "shortCode": "ABC123",  
        "passengerName": "Abebe Bikila",  
        "seatNumber": 5,  
        "qrCode": "data:image/png;base64,...",  
        "createdAt": "2026-02-19T10:01:00.000Z",  
        "trip": {  
            "id": "clx1trip010001234abc56de",  
            "origin": "Addis Ababa",  
            "destination": "Hawassa",  
            "departureTime": "2026-02-20T06:00:00.000Z",  
            "busType": "STANDARD",  
            "company": "Selam Bus"  
        },  
        "booking": {  
            "id": "clx1book010001234abc56de",  
            "totalAmount": 2120,  
            "bookedBy": "Abebe Bikila",  
            "passengerCount": 2  
        }  
    }  
}
```

Already Used Ticket

Response (200 OK):

```
{  
    "valid": false,  
    "error": "Ticket already used on 2026-02-20T06:30:00.000Z",  
    "ticket": {  
        "shortCode": "ABC123",  
        "passengerName": "Abebe Bikila",  
        "seatNumber": 5,  
        "usedAt": "2026-02-20T06:30:00.000Z"  
    }  
}
```

Mark Ticket Used: [PATCH /api/tickets/verify](#)

Request:

```
PATCH /api/tickets/verify HTTP/1.1  
Host: i-ticket.et  
Content-Type: application/json  
Cookie: next-auth.session-token=eyJhbGc...  
  
{  
    "ticketId": "clx1tick010001234abc56de"  
}
```

Response (200 OK):



```
{  
    "success": true,  
    "message": "Ticket marked as used successfully",  
    "ticket": {  
        "id": "clx1ticket010001234abc56de",  
        "passengerName": "Abebe Bikila",  
        "seatNumber": 5,  
        "shortCode": "ABC123",  
        "isUsed": true,  
        "usedAt": "2026-02-20T06:30:00.000Z"  
    }  
}
```

7. GPS Tracking Update (Driver Browser)

Endpoint: `POST /api/tracking/update`

Authentication: JWT session (driver/conductor assigned to trip) Rate Limit: 12 requests/min per user

Successful GPS Update

Request:

```
POST /api/tracking/update HTTP/1.1  
Host: i-ticket.et  
Content-Type: application/json  
Cookie: next-auth.session-token=eyJhbGc...  
  
{  
    "tripId": "clx1trip010001234abc56de",  
    "latitude": 7.0546,  
    "longitude": 38.4955,  
    "altitude": 1750.5,  
    "accuracy": 15.0,  
    "heading": 185.3,  
    "speed": 78.5,  
    "recordedAt": "2026-02-20T08:30:00.000Z"  
}
```

Response (200 OK):

```
{  
    "success": true,  
    "estimatedArrival": "2026-02-20T10:15:00.000Z"  
}
```

Invalid GPS Data

Response (400 Bad Request):



```
{  
    "error": "Invalid GPS data",  
    "details": [  
        {  
            "code": "too_small",  
            "minimum": -90,  
            "type": "number",  
            "inclusive": true,  
            "exact": false,  
            "message": "Number must be greater than or equal to -90",  
            "path": ["latitude"]  
        }  
    ]  
}
```

8. OsmAnd Background GPS

Endpoint: `GET /api/tracking/osmand`

Authentication: Trip tracking token (query parameter) **Rate Limit:** 12 requests/min per token **Response Format:** Plain text (not JSON)

Successful GPS Update

Request (from OsmAnd app):

```
GET /api/tracking/osmand?token=trk_abc123def456&lat=7.0546&lon=38.4955&altitude=1750&speed=21.8&bearing=185&timestamp=1708424400  
HTTP/1.1  
Host: i-ticket.et
```

Response (200 OK):

OK

Invalid Token

Response (200 OK) — returns 200 to prevent OsmAnd retry storms:

INVALID_TOKEN

Rate Limited

Response (200 OK):

RATE_LIMITED

9. Mark Passengers No-Show

Endpoint: `POST /api/company/trips/{tripId}/no-show`

Authentication: JWT session (COMPANY_ADMIN or SUPER_ADMIN) **Constraint:** Trip must be DEPARTED

Successful No-Show Marking

Request:



```
POST /api/company/trips/clx1trip010001234abc56de/no-show HTTP/1.1
Host: i-ticket.et
Content-Type: application/json
Cookie: next-auth.session-token=eyJhbGc...
{
  "passengerIds": [
    "clx1pass010001234abc56de",
    "clx1pass020001234abc56de"
  ]
}
```

Response (200 OK):

```
{
  "success": true,
  "noShowCount": 3,
  "releasedSeats": 2,
  "markedPassengers": [
    {
      "id": "clx1pass010001234abc56de",
      "name": "Abebe Bikila",
      "seatNumber": 5
    }
  ],
  "skippedPassengers": [
    {
      "id": "clx1pass020001234abc56de",
      "name": "Sara Bekele",
      "reason": "Already BOARDED"
    }
  ]
}
```

Trip Not Departed

Response (400 Bad Request):

```
{
  "error": "No-show marking is only allowed for DEPARTED trips (current: SCHEDULED)"
}
```

10. Replacement Ticket Sale

Endpoint: `POST /api/company/trips/{tripId}/replacement-ticket`

Authentication: JWT session (COMPANY_ADMIN or SUPER_ADMIN) Constraint: Trip DEPARTED + released seats available

Successful Replacement Sale

Request:



```
POST /api/company/trips/clx1trip010001234abc56de/replacement-ticket HTTP/1.1
Host: i-ticket.et
Content-Type: application/json
Cookie: next-auth.session-token=eyJhbGc...
{
  "passengerCount": 1
}
```

Response (200 OK):

```
{
  "success": true,
  "bookingId": "clx1book0200001234abc56de",
  "seatNumbers": [5],
  "shortCodes": ["GHI789"],
  "totalAmount": 1000,
  "releasedSeats": 1,
  "companyName": "Selam Bus",
  "route": "Addis Ababa → Hawassa",
  "vehicle": {
    "plateNumber": "3-12345",
    "sideNumber": "SB-001"
  }
}
```

Not Enough Released Seats

Response (400 Bad Request):

```
{
  "error": "Not enough released seats. Available: 1, requested: 3"
}
```

11. Security Controls Summary

Control	Endpoints
No Auth (Public)	Trip search, Registration
JWT Session Auth	Bookings, Payments, Ticket verify, GPS update, No-show, Replacement
Token Auth	OsmAnd GPS
HMAC-SHA256	TeleBirr callback
Rate Limiting (IP)	Registration (3/hr), Bookings (10/min), GPS (12/min)
Enhanced Rate Limiting	Payments (IP + user + per-booking)
Replay Protection	TeleBirr callback (ProcessedCallback table)
Row-Level Locking	Bookings (SELECT FOR UPDATE NOWAIT)
Transaction Timeouts	Bookings (10s), Payments (10s), No-show (10s), Replacement (15s)
Server-Side Recalculation	Payments (amount, commission, VAT)
Company Segregation	Ticket verify, No-show, Replacement
Zod Validation	GPS update, OsmAnd, Trip search, Ticket verify
Audit Logging	All mutation endpoints -> AdminLog table



End of Document 5.4.1 - API Request/Response Samples





D O C U M E N T

5.4.2

API Documentation

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Updated API Documentation

Document: 5.4.2 - Updated API Documentation Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform
 URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. API Architecture Overview

Property	Value
Framework	Next.js 14 App Router (file-based routing)
Base URL	https://i-ticket.et/api
Protocol	HTTPS only (TLS 1.2+ via Cloudflare)
API Versioning	None — single-version API at <code>/api/</code> base path
Data Format	JSON (<code>application/json</code>) for all request/response bodies
Authentication	Cookie-based JWT sessions (NextAuth.js)
Rate Limiting	Per-endpoint in-memory rate limiter (IP + user-based)
Total Route Files	145+ <code>route.ts</code> files across 20+ feature domains

2. Authentication Mechanism

2.1 Session-Based Authentication

All protected routes use NextAuth.js cookie-based JWT sessions. The session cookie is set on login and validated per-request.

Property	Value
Strategy	JWT (stateless)
Cookie name	<code>next-auth.session-token</code> (production: <code>__Secure-next-auth.session-token</code>)
Cookie flags	<code>httpOnly: true, secure: true, sameSite: 'lax'</code>
Max age	30 days
Token payload	<code>{ id, name, phone, role, companyId, staffRole }</code>

2.2 Auth Helper Functions

Authentication is enforced per-route using helpers from `src/lib/auth-helpers.ts`:

Helper	Required Role	Usage
<code>requireAuth()</code>	Any authenticated user	General protected routes
<code>requireRole(["CUSTOMER"])</code>	Specific role(s)	Role-restricted endpoints
<code>requireCompanyAdmin()</code>	<code>COMPANY_ADMIN</code> + <code>companyId</code>	Company management routes
<code>requireSuperAdmin()</code>	<code>SUPER_ADMIN</code>	Platform admin routes
<code>requireSalesPerson()</code>	<code>SALES_PERSON</code>	Sales portal routes
<code>requireFullAdmin()</code>	<code>COMPANY_ADMIN</code> (no supervisor)	Sensitive company settings

2.3 Alternative Auth Mechanisms

Mechanism	Used By	How
Cron secret	/api/cron/*	Authorization: Bearer <CRON_SECRET> header
OsmAnd token	/api/tracking/osmand	?token=<trackingToken> query parameter
SMS session	/api/bookings	smsSessionId in request body
Telegram secret	/api/telegram/webhook	Telegram-provided secret token header
TeleBirr HMAC	/api/payments/telebirr/callback	HMAC-SHA256 signature verification

2.4 Error Responses

Status	Response
401	{ "error": "Authentication required" }
403	{ "error": "Insufficient permissions" }
429	{ "error": "Too many requests. Please try again later.", "retryAfter": N }

3. Rate Limiting Configuration

In-memory rate limiter (`src/lib/rate-limit.ts`), keyed by IP address (`x-forwarded-for` or `x-real-ip` header).

Endpoint Category	Limit	Window	Key
Login	5 requests	15 min	Per phone + per IP
Registration	3 requests	1 hour	Per IP
Password reset	3 requests	1 hour	Per IP
Booking creation	10 requests	1 min	Per IP
Payment processing	5 requests	1 min	Per IP + per user
Payment (per booking)	3 requests	1 hour	Per booking ID
Support ticket	5 requests	1 hour	Per IP
Webhooks	100 requests	1 min	Per source
Vehicle management	20 requests	1 hour	Per IP
GPS tracking update	12 requests	1 min	Per user/token
Public tracking read	30 requests	1 min	Per IP
Default API	60 requests	1 min	Per IP

Rate-limited responses include `Retry-After` header with seconds until window reset.

4. Complete API Route Inventory

4.1 Authentication Routes (/api/auth/)

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/auth/[...nextauth]	None	LOGIN	NextAuth.js handler (login/logout/session)

POST	/api/auth/register	None	REGISTER	User registration
POST	/api/auth/forgot-password	None	FORGOT_PASSWORD	Request password reset
POST	/api/auth/reset-password	None	—	Complete password reset (token-based)
POST	/api/auth/force-change-password	Authenticated	—	Forced password change for staff

4.2 Public Trip Routes (/api/trips/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/trips	None	DEFAULT	Search available trips
GET	/api/trips/[tripId]	None	DEFAULT	Get single trip details
PATCH	/api/trips/[tripId]	COMPANY_ADMIN	DEFAULT	Update trip (own company)
GET	/api/trips/[tripId]/seats	None	DEFAULT	Get seat availability map
GET	/api/trips/[tripId]/messages	Authenticated	DEFAULT	Get trip broadcast messages
GET	/api/trips/[tripId]/export-gpx	Authenticated	DEFAULT	Export GPS trail as GPX file

4.3 Booking Routes (/api/bookings/)

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/bookings	Auth/SMS/Guest	CREATE_BOOKING	Create new booking
GET	/api/bookings	Authenticated	DEFAULT	List own bookings
GET	/api/bookings/[bookingId]	Owner/Admin	DEFAULT	Get booking details
PATCH	/api/bookings/[bookingId]	Owner/Admin	DEFAULT	Cancel booking

4.4 Payment Routes (/api/payments/)

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/payments	Owner/Guest	PROCESS_PAYMENT	Process payment for booking
POST	/api/payments/telebirr/callback	TeleBirr HMAC	WEBHOOK	TeleBirr payment callback
GET	/api/payments/telebirr/callback	None	DEFAULT	Callback endpoint health check

4.5 Ticket Routes (/api/tickets/)

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/tickets/verify	COMPANY_ADMIN/SUPER_ADMIN	DEFAULT	Verify ticket by short code
PATCH	/api/tickets/verify	COMPANY_ADMIN/SUPER_ADMIN	DEFAULT	Mark ticket as used + set BOARDED
GET	/api/tickets/verify/public	None	DEFAULT	Public QR scan verification page

4.6 GPS Tracking Routes (/api/tracking/)

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/tracking/update	Assigned driver/admin	12/min	Send GPS position from browser
GET	/api/tracking/osmand	Token-based	12/min	OsmAnd background GPS updates
POST	/api/tracking/generate-token	Driver/Admin	DEFAULT	Generate OsmAnd tracking token
GET	/api/tracking/[tripId]	None (public)	30/min	Get live bus position + history

GET	/api/tracking/fleet	COMPANY_ADMIN	DEFAULT	Company fleet positions map
GET	/api/tracking/active-trip	Authenticated driver	DEFAULT	Get driver's active trip

4.7 Booking Lookup Routes (/api/track/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/track/[code]	None (public)	DEFAULT	Look up booking by ID or ticket code
POST	/api/track/scan	Authenticated (staff)	DEFAULT	QR code scan lookup

4.8 City Routes (/api/cities/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/cities	None	DEFAULT	List all cities
GET	/api/cities/coordinates	None	DEFAULT	Get city coordinates for maps

4.9 Company Trip Management (/api/company/trips/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/trips	COMPANY_ADMIN	DEFAULT	List company trips (paginated)
POST	/api/company/trips	COMPANY_ADMIN	DEFAULT	Create trip
GET	/api/company/trips/[tripId]	COMPANY_ADMIN	DEFAULT	Get trip detail
PUT	/api/company/trips/[tripId]	COMPANY_ADMIN	DEFAULT	Update trip
DELETE	/api/company/trips/[tripId]	COMPANY_ADMIN	DEFAULT	Delete trip
POST	/api/company/trips/[tripId]/status	COMPANY_ADMIN	DEFAULT	Update trip status
GET	/api/company/trips/[tripId]/boarding-status	COMPANY_ADMIN	DEFAULT	Get boarding checklist
POST	/api/company/trips/[tripId]/no-show	COMPANY_ADMIN	DEFAULT	Mark passengers as no-show
POST	/api/company/trips/[tripId]/replacement-ticket	COMPANY_ADMIN	DEFAULT	Sell replacement ticket
GET	/api/company/trips/[tripId]/manifest	COMPANY_ADMIN	DEFAULT	Download passenger manifest
POST	/api/company/trips/[tripId]/manual-ticket	COMPANY_ADMIN (cashier)	DEFAULT	Manual ticket sale
POST	/api/company/trips/[tripId]/toggle-booking	COMPANY_ADMIN	DEFAULT	Halt/resume online booking
PATCH	/api/company/trips/[tripId]/auto-halt-setting	COMPANY_ADMIN	DEFAULT	Trip auto-halt override
POST	/api/company/trips/[tripId]/log	COMPANY_ADMIN	DEFAULT	Trip odometer log
POST	/api/company/trips/[tripId]/alert-response	COMPANY_ADMIN	DEFAULT	Respond to low-slot alert
POST	/api/company/trips/bulk	COMPANY_ADMIN	DEFAULT	Bulk create trips
POST	/api/company/trips/batch	COMPANY_ADMIN	DEFAULT	Batch trip operations
POST	/api/company/trips/import	COMPANY_ADMIN	DEFAULT	Import trips from CSV/Excel
GET	/api/company/trips/import/template	COMPANY_ADMIN	DEFAULT	Download import template
POST	/api/company/trips/import/validate	COMPANY_ADMIN	DEFAULT	Validate import data

4.10 Trip Templates (/api/company/trip-templates/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/trip-templates	COMPANY_ADMIN	DEFAULT	List trip templates

POST	/api/company/trip-templates	COMPANY_ADMIN	DEFAULT	Create trip template
GET	/api/company/trip-templates/[id]	COMPANY_ADMIN	DEFAULT	Get template
PUT	/api/company/trip-templates/[id]	COMPANY_ADMIN	DEFAULT	Update template
DELETE	/api/company/trip-templates/[id]	COMPANY_ADMIN	DEFAULT	Delete template

4.11 Staff Management (/api/company/staff/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/staff	COMPANY_ADMIN	DEFAULT	List company staff
POST	/api/company/staff	COMPANY_ADMIN	DEFAULT	Create staff member
GET	/api/company/staff/[staffId]	COMPANY_ADMIN	DEFAULT	Get staff member
PATCH	/api/company/staff/[staffId]	COMPANY_ADMIN	DEFAULT	Update staff member
DELETE	/api/company/staff/[staffId]	COMPANY_ADMIN	DEFAULT	Delete staff member

4.12 Vehicle & Fleet Management (/api/company/vehicles/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/vehicles	COMPANY_ADMIN	DEFAULT	List vehicles
POST	/api/company/vehicles	COMPANY_ADMIN	20/hr	Add vehicle
GET	/api/company/vehicles/[vehicleId]	COMPANY_ADMIN	DEFAULT	Get vehicle detail
PATCH	/api/company/vehicles/[vehicleId]	COMPANY_ADMIN	DEFAULT	Update vehicle
DELETE	/api/company/vehicles/[vehicleId]	COMPANY_ADMIN	DEFAULT	Delete vehicle
GET	/api/company/vehicles/[vehicleId]/maintenance-schedules	COMPANY_ADMIN	DEFAULT	List maintenance schedules
POST	/api/company/vehicles/[vehicleId]/maintenance-schedules	COMPANY_ADMIN	DEFAULT	Create maintenance schedule
GET	/api/company/vehicles/[vehicleId]/maintenance-schedules/[scheduleId]	COMPANY_ADMIN	DEFAULT	Get schedule detail
PATCH	/api/company/vehicles/[vehicleId]/maintenance-schedules/[scheduleId]	COMPANY_ADMIN	DEFAULT	Update schedule
DELETE	/api/company/vehicles/[vehicleId]/maintenance-schedules/[scheduleId]	COMPANY_ADMIN	DEFAULT	Delete schedule
GET	/api/company/vehicles/[vehicleId]/work-orders	COMPANY_ADMIN	DEFAULT	List vehicle work orders
POST	/api/company/vehicles/[vehicleId]/work-orders	COMPANY_ADMIN	DEFAULT	Create work order
GET	/api/company/vehicles/[vehicleId]/inspections	COMPANY_ADMIN	DEFAULT	List inspections
POST	/api/company/vehicles/[vehicleId]/inspections	COMPANY_ADMIN	DEFAULT	Create inspection
GET	/api/company/vehicles/[vehicleId]/fuel-entries	COMPANY_ADMIN	DEFAULT	List fuel entries
POST	/api/company/vehicles/[vehicleId]/fuel-entries	COMPANY_ADMIN	DEFAULT	Create fuel entry

4.13 Work Orders (/api/company/work-orders/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/work-orders	COMPANY_ADMIN	DEFAULT	List all work orders

POST	/api/company/work-orders	COMPANY_ADMIN	DEFAULT	Create work order
GET	/api/company/work-orders/[workOrderId]	COMPANY_ADMIN	DEFAULT	Get work order detail
PATCH	/api/company/work-orders/[workOrderId]	COMPANY_ADMIN	DEFAULT	Update work order
GET	/api/company/work-orders/[workOrderId]/messages	COMPANY_ADMIN	DEFAULT	Work order messages
POST	/api/company/work-orders/[workOrderId]/messages	COMPANY_ADMIN	DEFAULT	Send message
GET	/api/company/work-orders/[workOrderId]/parts	COMPANY_ADMIN	DEFAULT	List parts
POST	/api/company/work-orders/[workOrderId]/parts	COMPANY_ADMIN	DEFAULT	Add part
PATCH	/api/company/work-orders/[workOrderId]/parts/[partId]	COMPANY_ADMIN	DEFAULT	Update part
DELETE	/api/company/work-orders/[workOrderId]/parts/[partId]	COMPANY_ADMIN	DEFAULT	Delete part
GET	/api/company/work-orders/export	COMPANY_ADMIN	DEFAULT	Export work orders (Excel)

4.14 Fleet Analytics (/api/company/analytics/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/analytics/fleet-health	COMPANY_ADMIN	DEFAULT	Fleet risk scores and health gauge
GET	/api/company/analytics/risk-trends	COMPANY_ADMIN	DEFAULT	Historical risk trend data
GET	/api/company/analytics/cost-forecast	COMPANY_ADMIN	DEFAULT	Predicted maintenance costs
GET	/api/company/analytics/failure-timeline	COMPANY_ADMIN	DEFAULT	Upcoming predicted failures
GET	/api/company/analytics/vehicle-comparison	COMPANY_ADMIN	DEFAULT	Vehicle comparison metrics
GET	/api/company/analytics/maintenance-windows	COMPANY_ADMIN	DEFAULT	Optimal maintenance windows
GET	/api/company/analytics/route-wear	COMPANY_ADMIN	DEFAULT	Route-based wear analysis
GET	/api/company/analytics/compliance-calendar	COMPANY_ADMIN	DEFAULT	Compliance/inspection calendar
GET	/api/company/analytics/bookings	COMPANY_ADMIN	DEFAULT	Booking analytics
GET	/api/company/analytics/passengers	COMPANY_ADMIN	DEFAULT	Passenger analytics
GET	/api/company/analytics/revenue	COMPANY_ADMIN	DEFAULT	Revenue analytics
GET	/api/company/analytics/routes	COMPANY_ADMIN	DEFAULT	Route performance analytics

4.15 Company Reports (/api/company/reports/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/company/reports/maintenance	COMPANY_ADMIN	DEFAULT	Maintenance cost report
GET	/api/company/reports/maintenance/export	COMPANY_ADMIN	DEFAULT	Export maintenance (Excel)
GET	/api/company/reports/vehicle-tco	COMPANY_ADMIN	DEFAULT	Total Cost of Ownership
GET	/api/company/reports/downtime	COMPANY_ADMIN	DEFAULT	Vehicle downtime report
GET	/api/company/reports/fleet-analytics/export	COMPANY_ADMIN	DEFAULT	Export fleet analytics (Excel)
GET	/api/company/reports/staff-trips	COMPANY_ADMIN	DEFAULT	Staff trip report

4.16 Company Settings & Profile

Method	Endpoint	Auth	Rate Limit	Description
PATCH	/api/company/settings/auto-halt	COMPANY_ADMIN (full)	DEFAULT	Company-wide auto-halt toggle

GET	/api/company/profile	COMPANY_ADMIN	DEFAULT	Get company profile
PATCH	/api/company/profile	COMPANY_ADMIN	DEFAULT	Update company profile
GET	/api/company/stats	COMPANY_ADMIN	DEFAULT	Dashboard statistics
GET	/api/company/audit-logs	COMPANY_ADMIN	DEFAULT	Paginated audit log
GET	/api/company/audit-logs/download	COMPANY_ADMIN	DEFAULT	Download audit log (CSV)
GET	/api/company/messages	COMPANY_ADMIN	DEFAULT	Admin-to-company messages
POST	/api/company/messages	COMPANY_ADMIN	DEFAULT	Send message
POST	/api/company/messages/mark-read	COMPANY_ADMIN	DEFAULT	Mark messages read
GET	/api/company/messages/unread-count	COMPANY_ADMIN	DEFAULT	Unread message count

4.17 Super Admin Routes (/api/admin/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/admin/stats	SUPER_ADMIN	DEFAULT	Platform-wide statistics
GET	/api/admin/bookings	SUPER_ADMIN	DEFAULT	All bookings (paginated, filtered)
GET	/api/admin/bookings/export	SUPER_ADMIN	DEFAULT	Export bookings (CSV/Excel)
GET	/api/admin/companies	SUPER_ADMIN	DEFAULT	List companies
POST	/api/admin/companies	SUPER_ADMIN	DEFAULT	Create company
GET	/api/admin/companies/[companyId]	SUPER_ADMIN	DEFAULT	Get company detail
PATCH	/api/admin/companies/[companyId]	SUPER_ADMIN	DEFAULT	Update company
DELETE	/api/admin/companies/[companyId]	SUPER_ADMIN	DEFAULT	Delete company
POST	/api/admin/companies/[companyId]/setup-staff	SUPER_ADMIN	DEFAULT	Setup initial admin staff
GET	/api/admin/trips	SUPER_ADMIN	DEFAULT	All trips cross-company
GET	/api/admin/trips/[tripId]	SUPER_ADMIN	DEFAULT	Get any trip
PATCH	/api/admin/trips/[tripId]	SUPER_ADMIN	DEFAULT	Update any trip
GET	/api/admin/manifests	SUPER_ADMIN	DEFAULT	Cross-company manifests
GET	/api/admin/platform-staff	SUPER_ADMIN	DEFAULT	List platform staff
POST	/api/admin/platform-staff	SUPER_ADMIN	DEFAULT	Create platform staff
GET	/api/admin/platform-staff/[staffId]	SUPER_ADMIN	DEFAULT	Get platform staff
PATCH	/api/admin/platform-staff/[staffId]	SUPER_ADMIN	DEFAULT	Update platform staff
DELETE	/api/admin/platform-staff/[staffId]	SUPER_ADMIN	DEFAULT	Delete platform staff
GET	/api/admin/sales-persons	SUPER_ADMIN	DEFAULT	List sales persons
POST	/api/admin/sales-persons	SUPER_ADMIN	DEFAULT	Create sales person
GET	/api/admin/sales-persons/[id]	SUPER_ADMIN	DEFAULT	Get sales person
PATCH	/api/admin/sales-persons/[id]	SUPER_ADMIN	DEFAULT	Update sales person
DELETE	/api/admin/sales-persons/[id]	SUPER_ADMIN	DEFAULT	Delete sales person
POST	/api/admin/sales-persons/[id]/payout	SUPER_ADMIN	DEFAULT	Process commission payout
GET	/api/admin/audit-logs	SUPER_ADMIN	DEFAULT	Platform-wide audit log
GET	/api/admin/audit-logs/download	SUPER_ADMIN	DEFAULT	Download audit log (CSV)

GET	/api/admin/tax-reports	SUPER_ADMIN	DEFAULT	VAT/tax reports
GET	/api/admin/company-messages	SUPER_ADMIN	DEFAULT	Admin-to-company messages
POST	/api/admin/company-messages	SUPER_ADMIN	DEFAULT	Send message to company
POST	/api/admin/company-messages/[companyId]/mark-read	SUPER_ADMIN	DEFAULT	Mark messages read
GET	/api/admin/company-messages/unread-count	SUPER_ADMIN	DEFAULT	Unread count
GET	/api/admin/reports/platform-revenue	SUPER_ADMIN	DEFAULT	Platform revenue report

4.18 Admin Analytics (/api/admin/analytics/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/admin/analytics/revenue	SUPER_ADMIN	DEFAULT	Revenue analytics
GET	/api/admin/analytics/platform-revenue	SUPER_ADMIN	DEFAULT	Platform revenue breakdown
GET	/api/admin/analytics/monthly-trends	SUPER_ADMIN	DEFAULT	Monthly trend data
GET	/api/admin/analytics/budget-progress	SUPER_ADMIN	DEFAULT	Budget tracking
GET	/api/admin/analytics/top-companies	SUPER_ADMIN	DEFAULT	Top companies by revenue
GET	/api/admin/analytics/top-routes	SUPER_ADMIN	DEFAULT	Top routes by bookings
GET	/api/admin/analytics/sales-commissions	SUPER_ADMIN	DEFAULT	Sales commission data
GET	/api/admin/analytics/settlements	SUPER_ADMIN	DEFAULT	Settlement analytics

4.19 Cashier Portal (/api/cashier/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/cashier/my-trips	Cashier	DEFAULT	Cashier's assigned trips
GET	/api/cashier/trip/[tripId]	Cashier	DEFAULT	Trip details for cashier
POST	/api/cashier/trip/[tripId]/sell	Cashier	DEFAULT	Sell manual ticket (CASH)

4.20 Staff Portal (/api/staff/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/staff/my-trips	Staff	DEFAULT	Staff's assigned trips
PATCH	/api/staff/trip/[tripId]/status	Staff	DEFAULT	Update trip status
GET	/api/staff/work-orders	Staff	DEFAULT	Staff work orders
GET	/api/staff/work-orders/[workOrderId]	Staff	DEFAULT	Work order detail
PATCH	/api/staff/work-orders/[workOrderId]	Staff	DEFAULT	Update work order
GET	/api/staff/work-orders/[workOrderId]/messages	Staff	DEFAULT	Work order messages
POST	/api/staff/work-orders/[workOrderId]/messages	Staff	DEFAULT	Send message
POST	/api/staff/work-orders/[workOrderId]/field-repair	Staff	DEFAULT	Field repair report

4.21 Mechanic Portal (/api/mechanic/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/mechanic/work-orders	Mechanic	DEFAULT	Mechanic work orders
GET	/api/mechanic/work-orders/[workOrderId]	Mechanic	DEFAULT	Work order detail

PATCH	/api/mechanic/work-orders/[workOrderId]	Mechanic	DEFAULT	Update work order
GET	/api/mechanic/work-orders/[workOrderId]/messages	Mechanic	DEFAULT	Messages
POST	/api/mechanic/work-orders/[workOrderId]/messages	Mechanic	DEFAULT	Send message
GET	/api/mechanic/work-orders/[workOrderId]/parts	Mechanic	DEFAULT	Parts list
POST	/api/mechanic/work-orders/[workOrderId]/parts	Mechanic	DEFAULT	Add part

4.22 Finance Portal (/api/finance/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/finance/work-orders	Finance	DEFAULT	Finance work order view
GET	/api/finance/work-orders/[workOrderId]	Finance	DEFAULT	Work order detail
PATCH	/api/finance/work-orders/[workOrderId]	Finance	DEFAULT	Approve/update costs
GET	/api/finance/work-orders/[workOrderId]/messages	Finance	DEFAULT	Messages
POST	/api/finance/work-orders/[workOrderId]/messages	Finance	DEFAULT	Send message

4.23 Sales Portal (/api/sales/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/sales/dashboard	SALES_PERSON	DEFAULT	Dashboard statistics
GET	/api/sales/commissions	SALES_PERSON	DEFAULT	Commission history
GET	/api/sales/referrals	SALES_PERSON	DEFAULT	Referral tracking
GET	/api/sales/my-team	SALES_PERSON	DEFAULT	Recruited sub-agents
GET	/api/sales/recruiter-info	SALES_PERSON	DEFAULT	Recruiter information
GET	/api/sales/qr-code	SALES_PERSON	DEFAULT	Referral QR code
GET	/api/sales/my-qr-code	SALES_PERSON	DEFAULT	Personal QR code
GET	/api/sales/profile	SALES_PERSON	DEFAULT	Get profile
PATCH	/api/sales/profile	SALES_PERSON	DEFAULT	Update profile
PATCH	/api/sales/profile/payment	SALES_PERSON	DEFAULT	Update payment info
PATCH	/api/sales/password	SALES_PERSON	DEFAULT	Change password

4.24 Support (/api/support/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/support/tickets	Authenticated	DEFAULT	List own support tickets
POST	/api/support/tickets	Authenticated	CREATE_TICKET	Create support ticket
GET	/api/support/tickets/[ticketId]	Owner/Admin	DEFAULT	Get ticket detail
PATCH	/api/support/tickets/[ticketId]	Owner/Admin	DEFAULT	Update ticket

4.25 Notifications (/api/notifications/)

Method	Endpoint	Auth	Rate Limit	Description
GET	/api/notifications	Authenticated	DEFAULT	List notifications
GET	/api/notifications/count	Authenticated	DEFAULT	Unread count

GET	/api/notifications/grouped	Authenticated	DEFAULT	Grouped notifications
POST	/api/notifications/mark-all-read	Authenticated	DEFAULT	Mark all as read

4.26 Cron Jobs (/api/cron/)

Method	Endpoint	Auth	Rate Limit	Description
GET/POST	/api/cron/cleanup	CRON_SECRET	—	Expired booking/payment cleanup + trip lifecycle
GET	/api/cron/trip-reminders	CRON_SECRET	—	SMS departure reminders
GET	/api/cron/predictive-maintenance	CRON_SECRET	—	Daily AI risk snapshots
GET	/api/cron/telegram-cleanup	CRON_SECRET	—	Expired Telegram session cleanup

4.27 Webhooks & External

Method	Endpoint	Auth	Rate Limit	Description
POST	/api/telegram/webhook	Telegram secret	WEBHOOK	Telegram bot updates
POST	/api/sms/incoming	SMS gateway auth	WEBHOOK	Incoming SMS handler
POST	/api/sms/outgoing	Internal	—	Outgoing SMS sender
POST	/api/csp-report	None (browser)	—	CSP violation reports

5. Key Design Patterns

5.1 Company Segregation

Every company-scoped route enforces data isolation:

```
// Pattern used in every company API route
const trip = await prisma.trip.findFirst({
  where: {
    id: tripId,
    companyId: session.user.companyId // ← Mandatory filter
  }
})
```

Company admins can only access their own company's data. Super admins can access all data. This is enforced at the query level, not middleware.

5.2 Transaction Safety

All financial and booking mutations use:

```
transactionWithTimeout(async (tx) => {
  // SELECT FOR UPDATE NOWAIT – row-level locking
  // 10-second timeout – prevents deadlocks
  // Automatic rollback on any failure
}, 10000)
```

5.3 Input Validation

Every endpoint validates input with Zod schemas before processing:

```
const schema = z.object({
  tripId: z.string().cuid(),
  passengers: z.array(passengerSchema).min(1).max(5),
})
const data = schema.parse(await request.json())
```

5.4 Error Response Pattern

All API errors follow consistent structure:

Status Code	Usage	Response Body
400	Validation error	{ "error": "Description of validation failure" }
401	Not authenticated	{ "error": "Authentication required" }
403	Insufficient role	{ "error": "Insufficient permissions" }
404	Resource not found	{ "error": "Trip not found" }
409	Conflict (booking race)	{ "error": "Selected seats are no longer available" }
429	Rate limited	{ "error": "Too many requests...", "retryAfter": N }
500	Internal error	{ "error": "Internal server error" }

No internal details (stack traces, SQL errors, file paths) are ever exposed in production error responses.

5.5 Timezone Handling

All date comparisons use Ethiopia timezone utilities:

- `hasDepartedEthiopia()` — checks if departure time has passed (UTC+3)
- `getNowEthiopia()` — current time in Africa/Addis_Ababa
- `isTodayEthiopia()` — date comparison in local timezone

This prevents the 3-hour UTC offset on the AWS EC2 server from causing premature trip status changes.

6. Route Count Summary

Domain	Route Files	Endpoints
Authentication	5	5
Public Trips	5	7
Bookings	3	4
Payments	2	3
Tickets	2	3
GPS Tracking	6	6
Booking Lookup	2	2
Cities	2	2
Company Trips	17	~25
Company Templates	2	5
Company Staff	2	5
Company Vehicles	7	16
Company Work Orders	6	11

Company Analytics	12	12
Company Reports	6	6
Company Settings	8	10
Admin Routes	22	~33
Admin Analytics	8	8
Cashier	3	3
Staff	6	8
Mechanic	4	7
Finance	3	5
Sales	10	11
Support	2	4
Notifications	4	4
Cron	4	4
Webhooks/External	4	4
Total	~145+	~210+

End of Document 5.4.2 - Updated API Documentation



D O C U M E N T

5.4.3

API Types

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - API Types

Document: 5.4.3 - API Types Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Data Type Validation Strategy

All API inputs are validated using **Zod** (runtime schema validation library) before any business logic executes. TypeScript provides compile-time type safety. Prisma ORM provides database-level type enforcement.

Layer	Technology	Purpose
API Input	Zod schemas	Runtime validation of all request bodies and query params
Application	TypeScript	Compile-time type checking
Database	Prisma ORM	Type-safe query builder + migration enforcement
Output	JSON serialization	Consistent response shapes

2. Database Field Enumerations

The platform uses string fields with documented allowed values (enforced by application-layer Zod validation):

2.1 User Roles

Field	Allowed Values	Description
User.role	CUSTOMER, COMPANY_ADMIN, SUPER_ADMIN, SALES_PERSON	Primary user role
User.staffRole	ADMIN, DRIVER, CONDUCTOR, MANUAL_TICKETER, MECHANIC, FINANCE, SUPERVISOR	Company staff sub-role
User.staffStatus	AVAILABLE, ON_TRIP, ON_LEAVE	Staff availability

2.2 Trip Statuses

Field	Allowed Values	Description
Trip.status	SCHEDULED, DELAYED, BOARDING, DEPARTED, COMPLETED, CANCELLED	Trip lifecycle state
Trip.delayReason	TRAFFIC, BREAKDOWN, WEATHER, WAITING_PASSENGERS, OTHER	Reason for delay

Status Lifecycle: SCHEDULED → BOARDING → DEPARTED → COMPLETED ; can branch to DELAYED or CANCELLED from early states.

Editable States: SCHEDULED, DELAYED, BOARDING Read-Only States: DEPARTED, COMPLETED, CANCELLED (plus SOLD_OUT when availableSlots == 0)

2.3 Booking & Payment

Field	Allowed Values	Description
Booking.status	PENDING, PAID, CANCELLED, COMPLETED	Booking state
Payment.method	TELEBIRR, DEMO, CASH	Payment method
Payment.status	PENDING, SUCCESS, FAILED, TIMEOUT	Payment state
Payment.initiatedVia	WEB, SMS	Payment channel

Passenger.boardingStatus	PENDING , BOARDED , NO_SHOW	Boarding state
--------------------------	-----------------------------	----------------

2.4 Vehicle & Fleet

Field	Allowed Values	Description
Vehicle.status	ACTIVE , MAINTENANCE , INACTIVE	Vehicle operational state
Vehicle.busType	MINI , STANDARD , LUXURY	Vehicle class
Vehicle.fuelType	DIESEL , PETROL , CNG , ELECTRIC , GASOLINE , HYBRID	Fuel type
WorkOrder.taskType	PREVENTIVE , CORRECTIVE , INSPECTION , EMERGENCY	Work order type
WorkOrder.status	OPEN , IN_PROGRESS , BLOCKED , COMPLETED , CANCELLED	Work order state
WorkOrder.priority	1 (Low), 2 (Normal), 3 (High), 4 (Urgent)	Priority level
VehicleInspection.type	DAILY , WEEKLY , PRE_TRIP , POST_TRIP , ANNUAL , SAFETY	Inspection type
VehicleInspection.status	PASS , FAIL , PASS_WITH_DEFECTS	Inspection result
MaintenanceSchedule.taskType	PREVENTIVE , INSPECTION , SERVICE	Schedule type

2.5 Support & Sales

Field	Allowed Values	Description
SupportTicket.category	GENERAL , TECHNICAL , BOOKING , PAYMENT , ACCOUNT , FEEDBACK	Ticket category
SupportTicket.status	OPEN , IN_PROGRESS , RESOLVED , CLOSED	Ticket state
SupportTicket.priority	1 (Low), 2 (Medium), 3 (High), 4 (Urgent)	Priority
SalesCommission.status	PENDING , APPROVED , PAID	Commission payout state

2.6 Fuel & Logistics

Field	Allowed Values	Description
FuelEntry.fuelType	DIESEL , PETROL , CNG	Fuel type recorded
FuelEntry.paymentMethod	CASH , FUEL_CARD , TELEBIRR	Fuel payment method
TripLog.startFuelUnit	LITERS , PERCENTAGE	Fuel measurement unit
ManifestDownload.downloadType	AUTO_DEPARTED , AUTO_FULL_CAPACITY , MANUAL_COMPANY	Manifest trigger

3. Input Validation Schemas (Zod)

3.1 Authentication Schemas

Registration (`registerUserSchema`)

```
{  
  name: string, min 2 chars  
  phone: Ethiopian phone (09XXXXXXXX / 07XXXXXXXX, normalizes +251 prefix)  
  email: valid email (optional, or empty string)  
  password: string, 8-72 chars, requires uppercase + lowercase + digit  
  nationalId: string (optional)  
  nextOfKinName: string (optional)  
  nextOfKinPhone: Ethiopian phone (optional, or empty string)  
  referralCode: string (optional)  
  registerAsSalesPerson: boolean (optional)  
}
```

Login (`loginSchema`)

```
{  
  phone: Ethiopian phone  
  password: string, min 1 char  
}
```

Password Reset (`resetPasswordSchema`)

```
{  
  token: string, min 1 char  
  newPassword: string, 8-72 chars, requires uppercase + lowercase + digit  
}
```

3.2 Booking Schema (`createBookingSchema`)

```
{  
  tripId: string, min 1 char  
  passengers: array (1-10 items) of {  
    name: string, min 2 chars  
    nationalId: string (optional)  
    phone: Ethiopian phone  
    specialNeeds: string (optional)  
    isChild: boolean (optional)  
    pickupLocation: string (optional)  
    dropoffLocation: string (optional)  
  }  
  smsSessionId: string (optional, for SMS/guest flows)  
  selectedSeats: array of integers (optional, for manual seat selection)  
}
```

Business Rules Enforced:

- Max 5 passengers per booking (online)
- Max 3 children per booking
- Children must have an adult
- Company admins and super admins cannot book

3.3 Payment Schema (`processPaymentSchema`)

```
{  
  bookingId: string, min 1 char  
  method:    "TELEBIRR" | "DEMO"  
  phone:     Ethiopian phone (optional)  
}
```

3.4 Trip Schemas

Create Trip (`createTripSchema`)

```
{  
  origin:           string, min 2 chars  
  destination:    string, min 2 chars  
  route:           string (optional, nullable)  
  intermediateStops: string (optional, nullable) – JSON array  
  departureTime:   ISO 8601 datetime (must be future)  
  estimatedDuration: integer, 30-2880 (minutes)  
  distance:         integer, 1-5000 (km)  
  price:            number, positive, max 100000 (ETB)  
  busType:          string, min 2 chars  
  totalSlots:       integer, positive, max 100  
  hasWater:         boolean (default false)  
  hasFood:          boolean (default false)  
  driverId:         string, min 1 char  
  conductorId:     string, min 1 char  
  manualTicketerId: string (optional, nullable)  
  vehicleId:        string, min 1 char  
  defaultPickup:    string, max 200 chars (optional, nullable)  
  defaultDropoff:   string, max 200 chars (optional, nullable)  
  overrideStaffConflict: boolean (default false)  
  overrideVehicleConflict: boolean (default false)  
  vehicleOverrideReason: string (optional)  
}
```

Update Trip (`updateTripSchema`)

All fields from `createTripSchema` made optional (`.partial()`).

Trip Status Update (`statusUpdateSchema`)

```
{  
  status:           "SCHEDULED" | "DELAYED" | "BOARDING" | "DEPARTED" | "COMPLETED" | "CANCELLED"  
  notes:            string (optional)  
  delayReason:     "TRAFFIC" | "BREAKDOWN" | "WEATHER" | "WAITING_PASSENGERS" | "OTHER" (optional)  
  skipPreTripCheck: boolean (optional, for overriding vehicle risk warnings)  
  skipPreTripCheckReason: string, min 10 chars (required if skipPreTripCheck = true)  
}
```



Batch Trip Creation (`batchTripSchema`)

```
Base fields:  
{  
    origin, destination, estimatedDuration, distance, price, busType,  
    totalSlots, hasWater, hasFood, intermediateStops, defaultPickup,  
    defaultDropoff, driverId, conductorId, manualTicketerId, vehicleId,  
    createReturnTrips: boolean (default false)  
}  
  
Mode A – Same departure time for all dates:  
{  
    sameTimeForAll: true  
    dates: array of date strings (1-10)  
    departureTime: "HH:MM"  
    returnDepartureTime: "HH:MM" (optional)  
}  
  
Mode B – Individual times per date:  
{  
    sameTimeForAll: false  
    trips: array (1-10) of { date, time, returnTime? }  
}
```

3.5 Vehicle Schemas

Create Vehicle (`createVehicleSchema`)

```
{  
    plateNumber: Ethiopian plate format ("3-12345" or "AA 12345"), uppercased  
    sideNumber: alphanumeric with hyphens/spaces (optional), uppercased  
    make: string, 2-50 chars  
    model: string, 2-50 chars  
    year: integer, 1990 to current year + 1  
    busType: "MINI" | "STANDARD" | "LUXURY"  
    color: string, 2-30 chars (optional, nullable)  
    totalSeats: integer, 4-100 (range varies by busType)  
    status: "ACTIVE" | "MAINTENANCE" | "INACTIVE" (default "ACTIVE")  
    registrationExpiry: date string (optional)  
    insuranceExpiry: date string (optional)  
    lastServiceDate: date string (optional)  
    nextServiceDate: date string (optional)  
    currentOdometer: integer, >= 0 (optional, nullable)  
    fuelCapacity: number, positive (optional, nullable)  
    fuelType: "DIESEL" | "GASOLINE" | "ELECTRIC" | "HYBRID" (optional, nullable)  
}
```

Seat Range Validation by Bus Type:

- MINI: 4-20 seats
- STANDARD: 20-50 seats
- LUXURY: 30-60 seats

3.6 Staff Schema (`createStaffSchema`)

```
{  
    name: string, min 2 chars  
    phone: Ethiopian phone (09XXXXXXXX)  
    email: valid email (optional, or empty string)  
    password: string, min 8 chars, requires uppercase + lowercase + digit  
    staffRole: uppercase with underscores only (e.g., "DRIVER", "CONDUCTOR")  
    licenseNumber: string (optional)  
    employeeId: string (optional)  
}
```



3.7 GPS Tracking Schemas

Browser GPS Update (`updateSchema`)

```
{  
    tripId:      string, min 1 char  
    latitude:    number, -90 to 90  
    longitude:   number, -180 to 180  
    altitude:    number (optional, nullable)  
    accuracy:    number, >= 0 (optional, nullable)  
    heading:     number, 0-360 (optional, nullable)  
    speed:       number, >= 0 (optional, nullable)  
    recordedAt: ISO 8601 datetime  
}
```

OsmAnd GPS Update (`osmandSchema`)

```
Query parameters (all coerced from strings):  
{  
    token:      string, min 1 char  
    lat:        number, -90 to 90  
    lon:        number, -180 to 180  
    timestamp:  number (optional) – Unix epoch seconds  
    hdop:       number, >= 0 (optional)  
    altitude:   number (optional)  
    speed:      number, >= 0 (optional) – m/s, converted to km/h internally  
    bearing:    number, 0-360 (optional)  
}
```

3.8 Work Order Schema (`createWorkOrderSchema`)

```
{  
    vehicleId:      string  
    title:          string, 3-200 chars  
    taskType:        "PREVENTIVE" | "CORRECTIVE" | "INSPECTION" | "EMERGENCY"  
    scheduleId:     string (optional)  
    description:    string, 5-2000 chars  
    priority:       integer, 1-4 (default 2)  
    assignedStaffIds: array of strings (optional)  
    serviceProvider: string (optional)  
    scheduledDate: ISO 8601 datetime (optional)  
}
```

3.9 Vehicle Inspection Schema (`createInspectionSchema`)

```
{  
    inspectionType:  "DAILY" | "WEEKLY" | "PRE_TRIP" | "POST_TRIP" | "ANNUAL" | "SAFETY"  
    inspectedById:  string  
    checklistResults: Record<string, boolean | string | number>  
    status:          "PASS" | "FAIL" | "PASS_WITH_DEFECTS"  
    defectsFound:   array of strings (optional)  
    odometerReading: integer, positive (optional)  
    notes:           string (optional)  
}
```



3.10 Fuel Entry Schema (`createFuelEntrySchema`)

```
{  
    liters:          number, positive  
    costBirr:       number, positive  
    odometerReading: integer, positive  
    fuelType:        "DIESEL" | "PETROL" | "CNG" (default "DIESEL")  
    station:         string (optional)  
    city:            string (optional)  
    receiptNumber:   string (optional)  
    paymentMethod:   "CASH" | "FUEL_CARD" | "TELEBIRR" (optional)  
    recordedByUserId: string (optional)  
    recordedByName:  string (optional)  
}
```

3.11 Maintenance Schedule Schema

```
{  
    taskName:        string, 3-200 chars  
    taskType:        "PREVENTIVE" | "INSPECTION" | "SERVICE"  
    description:     string (optional)  
    intervalKm:      integer, positive (at least one interval required)  
    intervalDays:    integer, positive (at least one interval required)  
    priority:        integer, 1-4 (default 2)  
    estimatedDuration: integer, positive (optional) - minutes  
    estimatedCostBirr: number, positive (optional)  
    autoCreateWorkOrder: boolean (default true)  
}
```

3.12 Support Ticket Schema

```
{  
    name:      string, min 2 chars  
    email:     valid email  
    phone:     string (optional)  
    subject:   string, min 3 chars  
    message:   string, min 10 chars  
    category:  "GENERAL" | "TECHNICAL" | "BOOKING" | "PAYMENT" | "ACCOUNT" | "FEEDBACK" (optional)  
}
```

3.13 Trip Log Schemas

Start Readings

```
{  
    startOdometer: integer, >= 0  
    startFuel:     number, >= 0 (optional)  
    startFuelUnit: "LITERS" | "PERCENTAGE" (optional)  
    startNotes:    string, max 500 chars (optional)  
}
```

End Readings

```
{  
    endOdometer: integer, >= 0  
    endFuel:     number, >= 0 (optional)  
    endNotes:    string, max 500 chars (optional)  
}
```

3.14 No-Show Marking Schema

```
{  
    passengerIds: array of strings (CUID passenger IDs)  
}
```

3.15 Replacement Ticket Schema

```
{  
    passengerCount: integer, 1-10 (default 1)  
}
```

3.16 Admin Schemas

Create Company (`CreateCompanySchema`)

```
{  
    companyName: string, min 2 chars  
    companyPhone: Ethiopian phone (09XXXXXXXX)  
    companyEmail: valid email  
    address: string (optional)  
    bankName: string (optional)  
    bankAccount: string (optional)  
    bankBranch: string (optional)  
    adminName: string, min 2 chars  
    adminPhone: Ethiopian phone  
    adminEmail: valid email  
}
```

Company Status Update (`CompanyStatusSchema`)

```
{  
    companyId: string, min 1 char  
    isActive: boolean  
    reason: string, 10-500 chars (trimmed)  
}
```

3.17 Query/Filter Schemas

Pagination (`paginationSchema`)

```
{  
    page: string (default "1") → parsed to integer >= 1  
    limit: string (default "20") → parsed to integer, clamped 1-100  
}
```

Trip Search (`searchTripsSchema`)

```
{  
    origin: string (optional)  
    destination: string (optional)  
    date: string (optional)  
    page: string (default "1")  
    limit: string (default "20")  
}
```



Admin Bookings (`adminBookingsQuerySchema`)

```
{  
  page:      integer, >= 1 (default 1)  
  limit:     integer, 1-100 (default 10)  
  status:    "PAID" | "PENDING" | "CANCELLED" | "ALL" (default "ALL")  
  companyId: string (optional)  
  startDate: string (optional)  
  endDate:   string (optional)  
  search:    string (optional)  
}
```

Work Order Query (`workOrderQuerySchema`)

```
{  
  status:    "OPEN" | "IN_PROGRESS" | "BLOCKED" | "COMPLETED" | "CANCELLED" (optional)  
  priority: string → integer 1-4 (optional)  
  vehicleId: string (optional)  
  workType:  "PREVENTIVE" | "CORRECTIVE" | "INSPECTION" | "EMERGENCY" (optional)  
  page:      string → integer >= 1 (optional)  
  limit:     string → integer 1-100 (optional)  
}
```

4. TypeScript Type Definitions

4.1 Authentication Session Type

```
interface Session {  
  user: {  
    id: string  
    name: string  
    email?: string  
    phone: string  
    role: "CUSTOMER" | "COMPANY_ADMIN" | "SUPER_ADMIN" | "SALES_PERSON"  
    companyId: string | null  
    companyName: string | null  
    companyLogo: string | null  
    staffRole: string | null  
    profilePicture: string | null  
    nationalId: string | null  
    mustChangePassword?: boolean  
    platformStaffRole?: string  
    platformStaffDepartment?: string  
    platformStaffPermissions?: Record<string, boolean>  
  }  
}
```

4.2 JWT Token Payload

```
interface JWT {  
  id: string  
  role: string  
  phone: string  
  companyId: string | null  
  companyName: string | null  
  companyLogo: string | null  
  staffRole: string | null  
  profilePicture: string | null  
  nationalId: string | null  
  mustChangePassword?: boolean  
  platformStaffRole?: string  
  platformStaffDepartment?: string  
  platformStaffPermissions?: Record<string, boolean>  
}
```

4.3 Commission Types

```
interface CommissionBreakdown {  
  baseCommission: number // 5% of totalAmount  
  vat: number // 15% of baseCommission  
  totalCommission: number // baseCommission + vat  
}  
  
// Constants  
COMMISSION_RATE = 0.05 // 5%  
VAT_RATE = 0.15 // 15% Ethiopian VAT  
TELEBIRR_FEE_RATE = 0.005 // 0.5% (absorbed by i-Ticket)
```

4.4 TeleBirr Payment Types

```
interface TelebirrCallbackPayload {  
  transactionId: string  
  outTradeNo: string // booking ID  
  status: string // "SUCCESS" | "FAILED"  
  amount: string  
  currency: string // "ETB"  
  timestamp: string  
  signature: string  
}
```

4.5 Trip Status Types

```
type FinalTripStatus = "DEPARTED" | "COMPLETED" | "CANCELLED"  
type EditableTripStatus = "SCHEDULED" | "DELAYED" | "BOARDING"  
type TripStatus = FinalTripStatus | EditableTripStatus
```



5. Common Response Structures

5.1 Standard Error Response

```
{ "error": "Human-readable error message" }
```

Status codes: 400, 401, 403, 404, 409, 429, 500

5.2 Validation Error Response

```
{
  "error": "Validation error",
  "details": [
    { "code": "too_small", "minimum": 2, "path": ["name"], "message": "String must contain at least 2 character(s)" }
  ]
}
```

5.3 Paginated List Response

```
{
  "<items>": [...],
  "pagination": {
    "page": 1,
    "limit": 20,
    "total": 150,
    "totalPages": 8
  }
}
```

Field name varies by endpoint: `bookings`, `trips`, `workOrders`, `staff`, etc.

5.4 Rate Limit Response (HTTP 429)

```
{
  "error": "Too many requests. Please try again later.",
  "retryAfter": 45
}
```

Includes `Retry-After: 45` header.

6. Ethiopian Phone Number Format

All phone number fields use a custom Zod validator that normalizes multiple input formats:

Input Format	Normalized To	Valid
0912345678	0912345678	Yes
0712345678	0712345678	Yes
+251912345678	0912345678	Yes
251912345678	0912345678	Yes

912345678	Invalid	No
1234567890	Invalid	No

Final validation regex: `/^0[79]\d{8}$/` (10 digits, starting with 09 or 07)

7. Ethiopian License Plate Format

Vehicle plate numbers are validated with a custom pattern:

Format	Example	Valid
N-NNNNN	3-12345	Yes
LL NNNNN	AA 12345	Yes
LLNNNNN	AA12345	Yes
Random text	ABCXYZ	No

All plate numbers are converted to uppercase before storage.

8. Sensitive Field Classification

Fields classified by sensitivity level for audit purposes:

Sensitivity	Fields	Protection
Critical	User.password	bcrypt hash (12 rounds), never returned in API responses
Critical	NEXTAUTH_SECRET , TELEBIRR_APP_KEY	Environment variables, validated at startup
High	User.nationalId , Company.bankAccount	Stored plaintext; recommended for field-level encryption
High	Password reset tokens	bcrypt-hashed before storage, time-limited (1 hour)
Medium	User.phone , User.email	Validated format, used for identification
Medium	Trip.trackingToken	Crypto-random (32 bytes hex), one-per-trip
Low	Passenger.name , Booking.totalAmount	Standard business data

End of Document 5.4.3 - API Types



D O C U M E N T

5.4.4

API Endpoints and Functionality

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - API Endpoints and Functionality

Document: 5.4.4 - API Endpoints and Functionality Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Overview

This document describes the functional behavior of each API endpoint group — what business logic executes, what side effects occur, and what security controls are enforced. For request/response schemas, see Document 5.4.3 (API Types). For sample payloads, see Document 5.4.1 (API Request/Response Samples).

Base URL: <https://i-ticket.et/api>

2. Authentication Endpoints

2.1 POST /api/auth/[...nextauth] — Login

Functionality:

1. Receives phone + password credentials
2. Looks up user by normalized phone number
3. Compares password with bcrypt hash using timing-safe comparison
4. Checks `mustChangePassword` flag (forces redirect for staff first login)
5. Creates JWT session token with minimal claims (id, role, companyId, staffRole)
6. Sets `httpOnly`, `secure`, `sameSite: lax` cookie (30-day expiry)

Security Controls: Rate limited (5/15min per phone, 10/15min per IP). No user enumeration — same error message for wrong phone or wrong password.

2.2 POST /api/auth/register — Registration

Functionality:

1. Validates input with Zod (name, phone, email, password)
2. Password strength check: 8-72 chars, uppercase + lowercase + digit
3. Checks for duplicate phone number
4. Hashes password with bcrypt (12 salt rounds)
5. Creates User with role `CUSTOMER` (or `SALES_PERSON` if referral code provided)
6. If sales person registration: validates recruiter referral code, creates SalesPerson record with referral chain

Security Controls: Rate limited (3/hr per IP). Phone format validated. Password never stored in plaintext.

2.3 POST /api/auth/forgot-password — Request Reset

Functionality:

1. Receives phone number
2. Generates crypto-random 32-byte token
3. Hashes token with bcrypt before storage (prevents DB-leaked tokens from being usable)
4. Stores hashed token with 1-hour expiry
5. Sends SMS with reset link containing raw token

Security Controls: Rate limited (3/hr per IP). Same response regardless of phone existence (prevents enumeration).



2.4 POST /api/auth/reset-password — Complete Reset

Functionality:

1. Receives token + new password
2. Looks up user with non-expired reset token
3. Compares received token against stored bcrypt hash
4. Updates password (new bcrypt hash, 12 rounds)
5. Clears reset token fields

Security Controls: Token is one-time use. Password strength enforced. Token expires after 1 hour.

2.5 POST /api/auth/force-change-password

Functionality: Allows staff members with `mustChangePassword: true` to set a new password on first login. Clears the forced-change flag after successful update.

3. Booking Endpoints

3.1 POST /api/bookings — Create Booking

Functionality:

1. Validates passenger data (1-10 passengers, max 3 children with adult required)
2. Identifies user: NextAuth session, SMS session, or guest (creates guest user from first passenger's phone)
3. Blocks company admins and super admins from booking
4. Opens transaction with 10-second timeout
5. Acquires row lock on trip (`SELECT FOR UPDATE NOWAIT`) — prevents race conditions
6. Validates trip: must be `SCHEDULED`, `BOARDING`, or `DELAYED`; must have enough available slots
7. Assigns seat numbers (manual selection or auto-assign)
8. Creates Booking (status `PENDING`), Passenger records, Payment record (status `PENDING`)
9. Decrements `Trip.availableSlots` atomically
10. Checks auto-halt threshold: if `availableSlots <= 10` and auto-halt enabled, sets `bookingHalted = true`
11. Logs `BOOKING_CREATED` to AdminLog

Side Effects: Creates Notification for user. Starts 10-minute payment countdown.

Security Controls: Rate limited (10/min). Row-level DB locking. Server-side amount calculation. Company admin/super admin booking blocked. Transaction timeout prevents deadlocks.

3.2 GET /api/bookings — List Own Bookings

Functionality: Returns authenticated user's bookings with optional status filter. Includes trip, passenger, payment, and ticket details. Users can only see their own bookings.

3.3 PATCH /api/bookings/[bookingId] — Cancel Booking

Functionality:

1. Verifies ownership (user's booking or admin)
2. Only `PENDING` bookings can be cancelled
3. Opens transaction: cancels booking, releases slots (`availableSlots += passengerCount`)
4. If trip was auto-halted and slots now above threshold, resumes booking

4. Payment Endpoints

4.1 POST /api/payments — Process Payment

Functionality:

1. Validates booking ownership
2. Checks 10-minute payment window (rejects expired bookings)
3. **Server-side amount recalculation:** recomputes total from trip price × passengers + commission + VAT (never trusts client amount)
4. For DEMO mode: instantly marks payment SUCCESS
5. For TELEBIRR mode: initiates TeleBirr API call with HMAC-SHA256 signed request
6. On success: marks Booking PAID, generates QR-code Tickets (unique short codes), creates SalesCommission records
7. Sends SMS confirmation with ticket short codes to each passenger

Security Controls: Rate limited (5/min per IP, 3/booking/hr). Amount recalculated server-side. Payment replay protection via status check. Transaction-safe with row locking.

4.2 POST /api/payments/telebirr/callback — TeleBirr Webhook

Functionality (5-gate verification):

1. **Gate 1 — Hash Check:** Computes SHA-256 hash of raw payload
2. **Gate 2 — Replay Protection:** Checks `ProcessedCallback` table for duplicate callback hash
3. **Gate 3 — HMAC Verification:** `crypto.timingSafeEqual()` with `TELEBIRR_APP_KEY`
4. **Gate 4 — Timestamp Check:** Rejects callbacks older than 5 minutes
5. **Gate 5 — IP Whitelist (optional):** Checks `TELEBIRR_WEBHOOK_IPS` if configured
6. On SUCCESS: updates Payment, marks Booking PAID, generates Tickets, sends SMS
7. On FAILURE: marks Payment FAILED, cancels Booking, releases slots

Security Controls: HMAC-SHA256 signature, timing-safe comparison, replay protection, timestamp validation, optional IP whitelist. All writes in transaction.

5. Ticket Verification Endpoints

5.1 POST /api/tickets/verify — Verify Ticket

Functionality:

1. Receives ticket short code (case-insensitive)
2. Looks up ticket with booking, trip, and passenger data
3. Validates: booking must be PAID, ticket must not be already used
4. Enforces company segregation (verifier's company must match trip's company)
5. Returns ticket validity + passenger details

Security Controls: Company segregation. Case-insensitive lookup. AdminLog entry.

5.2 PATCH /api/tickets/verify — Mark Ticket Used

Functionality:

1. Sets `ticket.isUsed = true`, `ticket.usedAt = now`
2. Auto-sets matching passenger `boardingStatus = "BOARDED"`
3. If passenger was previously marked `NO_SHOW`, returns warning (late arrival edge case)
4. Logs `TICKET_USED` to AdminLog

6. GPS Tracking Endpoints

6.1 POST /api/tracking/update — Browser GPS

Functionality:

1. Validates GPS coordinates (lat -90/90, lon -180/180)
2. Verifies user is assigned driver/conductor on the trip, or is admin
3. Trip must be in `DEPARTED` status
4. Delegates to shared `processPositionUpdate()`:
 - o Creates `TripPosition` record (GPS history)
 - o Updates `Trip.lastLatitude`, `lastLongitude`, `lastSpeed`, `lastPositionAt`
 - o Updates `Vehicle.lastLatitude`, `lastLongitude`, `lastPositionAt`
 - o Calculates ETA using Haversine distance + 1.3 winding factor
5. Returns estimated arrival time

Security Controls: Rate limited (12/min). Auth required. Trip assignment verified.

6.2 GET /api/tracking/osmand — OsmAnd Background GPS

Functionality: Same as browser GPS but uses token-based auth for headless OsmAnd app. Returns plain text status codes (OsmAnd stops retrying on non-200, so all responses are HTTP 200 with text status).

Security Controls: Token validated against `Trip.trackingToken`. Rate limited (12/min). Token is 256-bit random hex.

6.3 GET /api/tracking/[tripId] — Public Bus Position

Functionality: Returns current bus position, speed, ETA, and GPS trail history for departed trips. Available to all passengers for live tracking.

Security Controls: Rate limited (30/min per IP). No sensitive data exposed (no driver phone, no passenger info).

6.4 GET /api/tracking/fleet — Company Fleet Map

Functionality: Returns all currently DEPARTED trips with GPS data for the authenticated company. Includes driver info, vehicle info, occupancy percentage, and tracking status (live/stale/off).

Security Controls: Company segregation enforced. Only company admin access.

7. No-Show Management Endpoints

7.1 GET /api/company/trips/[tripId]/boarding-status

Functionality: Returns all passengers for a trip with their boarding status (PENDING/BOARDED/NO_SHOW), seat numbers, pickup/dropoff locations, ticket codes, and whether they're replacement passengers. Includes summary counts.

Security Controls: Company segregation. Only BOARDING or DEPARTED trips.

7.2 POST /api/company/trips/[tripId]/no-show — Mark No-Shows

Functionality:

1. Trip must be `DEPARTED` (intermediate stops mean passengers may board later at stops)
2. Validates passenger IDs belong to this trip
3. For each passenger:
 - o Skips if already `NO_SHOW` (idempotent)
 - o Skips if `BOARDED` (ticket already scanned)
 - o Skips if ticket already used

- Sets `boardingStatus = "NO_SHOW"`
- 4. Updates `Trip.noShowCount` and `Trip.releasedSeats` counters in transaction
- 5. Logs `PASSENGER_NO_SHOW` to AdminLog

Security Controls: Trip status guard (DEPARTED only). Company segregation. Ticket-used check. Idempotent. Transaction-safe.

7.3 POST /api/company/trips/[tripId]/replacement-ticket — Sell Replacement

Functionality:

1. Trip must be `DEPARTED`
2. Checks `releasedSeats >= passengerCount` (can't sell more replacements than no-shows minus already replaced)
3. Opens transaction:
 - Creates Booking (`isReplacement = true`, `replacedPassengerId` for audit)
 - Creates Passenger records with `boardingStatus = "BOARDED"` immediately
 - Auto-assigns seat numbers from no-show passengers' seats
 - Creates Payment (CASH, SUCCESS), Tickets with QR codes
 - Updates `Trip.replacementsSold` counter
 - Recalculates `Trip.releasedSeats = noShowCount - replacementsSold`
4. Commission = 0 (replacement sales generate no platform commission)
5. Logs `REPLACEMENT_TICKET_SALE` to AdminLog

Security Controls: Company segregation. Released seat cap. Staff/cashier only (not online). CASH only. `availableSlots` never modified. Transaction-safe with row locking.

8. Company Trip Management Endpoints

8.1 POST /api/company/trips — Create Trip

Functionality:

1. Validates all fields (origin, destination, departure time, driver, conductor, vehicle)
2. Checks staff/vehicle availability (no scheduling conflicts unless override flags set)
3. Vehicle risk check: if risk score $>= 85$, requires pre-trip inspection or override with reason
4. Creates trip with `SCHEDULED` status, sets `availableSlots = totalSlots`
5. Logs `TRIP_CREATED` to AdminLog

Security Controls: Company segregation. Staff conflict detection. Vehicle risk warnings.

8.2 POST /api/company/trips/[tripId]/status — Update Trip Status

Functionality:

1. Validates status transition is valid (e.g., can't go COMPLETED \rightarrow SCHEDULED)
2. On `DEPARTED`:
 - Sets `actualDepartureTime`, activates GPS tracking
 - Auto-generates manifest if not already generated
 - Updates assigned staff status to `ON_TRIP`
3. On `COMPLETED`:
 - Sets `actualArrivalTime`, deactivates GPS tracking
 - Resets staff status to `AVAILABLE`
 - Shows odometer log popup for driver
4. On `CANCELLED`: Releases all unpaid booking slots, sends notifications
5. Pre-trip safety check: if vehicle risk score $>= 70$, warns; $>= 85$, requires recent inspection or explicit override

Security Controls: Status transition validation. Company segregation. Pre-trip safety checks. AdminLog entries.

8.3 POST /api/company/trips/[tripId]/manual-ticket — Manual Sale

Functionality: Creates instant PAID booking with CASH payment. Used by staff/cashiers at bus stations. No slot restriction (can sell to 0 available slots — manual ticketing is never blocked). Commission = 0.

8.4 POST /api/company/trips/[tripId]/toggle-booking — Halt/Resume

Functionality: Toggles `Trip.bookingHalted` flag. When halted, online booking is blocked but manual ticketing continues. Used for crowd control or when trip is nearly full.

9. Vehicle & Fleet Management Endpoints

9.1 CRUD /api/company/vehicles/

Functionality: Full vehicle lifecycle management. License plate validation (Ethiopian format). Auto-creates `VehicleDowntime` records when status changes to/from `MAINTENANCE`. Tracks odometer, fuel capacity, service dates, insurance/registration expiry.

9.2 Maintenance Schedules

Functionality: Recurring maintenance task definitions with kilometer and/or day intervals. Can auto-create work orders when thresholds are met. Tracks estimated duration and cost.

9.3 Work Orders

Functionality: Full work order lifecycle (OPEN → IN_PROGRESS → COMPLETED). Multi-role collaboration: admin creates, mechanic executes, finance approves costs. Supports parts tracking, messaging, and Excel export.

9.4 Vehicle Inspections

Functionality: Pre-trip, post-trip, daily, weekly, annual, and safety inspections. Checklist-based with pass/fail/defects. Links to trip safety checks (risk score integration).

9.5 Fuel Entries

Functionality: Fuel consumption logging with odometer-based efficiency calculation (L/100km). Supports multiple fuel types and payment methods.

10. Fleet Analytics Endpoints

All analytics endpoints: `GET /api/company/analytics/*`

Endpoint	Functionality
<code>fleet-health</code>	Aggregated fleet risk scores, health gauge (0-100), vehicle-level risk breakdown
<code>risk-trends</code>	Historical risk score snapshots from <code>VehicleRiskHistory</code> (daily cron snapshots)
<code>cost-forecast</code>	Predicted maintenance costs based on historical spending patterns and scheduled tasks
<code>failure-timeline</code>	Upcoming predicted component failures based on mileage, age, and maintenance history
<code>vehicle-comparison</code>	Side-by-side metrics for selected vehicles (cost, downtime, utilization)
<code>maintenance-windows</code>	Optimal scheduling windows based on trip schedules and vehicle availability
<code>route-wear</code>	Route-based wear analysis correlating trip routes with vehicle maintenance needs



compliance-calendar	Upcoming inspection due dates, registration/insurance expiries, overdue items
bookings	Booking volume, conversion rates, channel breakdown (web/SMS/Telegram)
passengers	Passenger demographics, repeat customer rate, average group size
revenue	Revenue trends, daily/weekly/monthly aggregates, route profitability
routes	Route performance: occupancy rates, cancellation rates, popular routes

Security Controls: All require `COMPANY_ADMIN` auth. Data scoped to authenticated company only.

11. Company Reports Endpoints

Endpoint	Functionality	Export
<code>reports/maintenance</code>	Maintenance costs by vehicle, category, time period	Excel
<code>reports/vehicle-tco</code>	Total Cost of Ownership per vehicle (purchase + maintenance + fuel + downtime)	—
<code>reports/downtime</code>	Vehicle downtime hours/days, causes, impact on operations	—
<code>reports/fleet-analytics/export</code>	Comprehensive fleet analytics data dump	Excel
<code>reports/staff-trips</code>	Staff trip assignments, completion rates, performance	—

12. Super Admin Endpoints

12.1 GET `/api/admin/stats` — Platform Dashboard

Functionality: Returns comprehensive platform-wide statistics:

- User counts (total, customers, company admins, guests, new today)
- Company counts (active, inactive)
- Trip counts (by status)
- Booking counts (total, paid, pending, cancelled, today)
- Revenue (total, today, yesterday, this week, this month, commissions, VAT)
- Channel breakdown (web, SMS, Telegram)
- Payment method breakdown (TeleBirr, demo, cash)
- Business insights (average booking value, cancellation rate, success rate, peak hours)
- Recent bookings list

12.2 Company Management (`/api/admin/companies`)

Functionality: Create, read, update, deactivate companies. Company creation auto-creates admin user with temporary password. Deactivation requires written reason (10-500 chars) and logs to AdminLog.

12.3 Admin Analytics (`/api/admin/analytics/*`)

Endpoint	Functionality
<code>revenue</code>	Platform-wide revenue analytics
<code>platform-revenue</code>	Commission + VAT breakdown, TeleBirr fee impact
<code>monthly-trends</code>	Month-over-month growth metrics
<code>budget-progress</code>	Revenue vs targets

<code>top-companies</code>	Companies ranked by booking volume/revenue
<code>top-routes</code>	Routes ranked by popularity/revenue
<code>sales-commissions</code>	Sales agent commission tracking
<code>settlements</code>	Company settlement (payout) analytics

13. Portal Endpoints

13.1 Cashier Portal (`/api/cashier/`)

Functionality: View assigned trips, sell manual tickets (CASH only), view trip passenger lists. Can sell tickets even when `availableSlots = 0` (manual ticketing never blocked).

13.2 Staff Portal (`/api/staff/`)

Functionality: View assigned trips, update trip status (board, depart, complete), manage work orders assigned to them, submit field repair reports.

13.3 Mechanic Portal (`/api/mechanic/`)

Functionality: View and update assigned work orders, track parts, communicate with admin/finance via work order messages.

13.4 Finance Portal (`/api/finance/`)

Functionality: Review work order costs, approve/reject expenditures, view financial summaries of maintenance spending.

13.5 Sales Portal (`/api/sales/`)

Functionality: View commission history, referral tracking, team management (recruited sub-agents), QR code generation for referral links, profile/payment info management.

14. Cron Job Endpoints

14.1 POST `/api/cron/cleanup` — Automated Cleanup

Runs: Every 5 minutes (PM2 cron or external scheduler)

Operations (in order):

1. Delete expired SMS sessions
2. Timeout SMS payments older than 5 minutes → cancel booking, release slots, send SMS
3. Cancel stale PENDING bookings older than 10 minutes → release slots
4. Auto-DELAYED: SCHEDULED trips 30+ minutes past departure time
5. Auto-DEPARTED: SCHEDULED/BOARDING/DELAYED trips past departure (DELAYED waits 1 hour extra)
6. Auto-COMPLETED: DEPARTED trips past `estimatedDuration + 2 hour` buffer
7. Auto-CANCELLED: SCHEDULED/BOARDING trips 24+ hours past with zero bookings
8. Reset staff status: drivers/conductors with no active trips → `AVAILABLE`
9. Purge GPS positions: delete `TripPosition` records >7 days old for COMPLETED/CANCELLED trips

Security Controls: Requires `CRON_SECRET` bearer token. All timezone calculations use Ethiopia time.

14.2 GET `/api/cron/trip-reminders` — SMS Reminders

Runs: Hourly. Sends SMS reminders to passengers with upcoming departures.

14.3 GET /api/cron/predictive-maintenance — AI Risk Snapshots

Runs: Daily. Calculates vehicle risk scores and stores in `VehicleRiskHistory` for trend analysis.

14.4 GET /api/cron/telegram-cleanup — Session Cleanup

Runs: Every 30 minutes. Deletes expired Telegram bot sessions (30-minute timeout).

15. Webhook Endpoints

15.1 POST /api/telegram/webhook

Functionality: Receives all Telegram bot updates (messages, callback queries). Routes to appropriate handler: `/start`, `/book`, `/mytickets`, `/whereismybus`, `/help`, `/cancel`. Manages multi-step booking wizard with session state.

15.2 POST /api/sms/incoming

Functionality: Handles incoming SMS messages for SMS-based booking flow.

15.3 POST /api/csp-report

Functionality: Receives Content-Security-Policy violation reports from browsers. Logs violations to PM2/stdout for monitoring. Returns HTTP 204 (no content).

16. Cross-Cutting Concerns

16.1 Company Segregation (All /api/company/* Routes)

Every company-scoped query includes `companyId` from the authenticated session:

```
WHERE companyId = session.user.companyId
```

This is enforced at the query level in every route handler, not at a middleware layer. Super admins bypass this filter.

16.2 Audit Logging

All sensitive operations create `AdminLog` entries:

Action	Logged Data
<code>BOOKING_CREATED</code>	bookingId, tripId, passengerCount, amount
<code>PAYMENT_PROCESSED</code>	paymentId, bookingId, method, amount
<code>TICKET_VERIFIED</code>	ticketId, shortCode, tripId
<code>TICKET_USED</code>	ticketId, passengerId
<code>TRIP_STATUS_CHANGED</code>	tripId, from status, to status
<code>PASSENGER_NO_SHOW</code>	tripId, passengerIds, counts
<code>REPLACEMENT_TICKET_SALE</code>	tripId, bookingId, seatNumbers, amount
<code>COMPANY_DEACTIVATED</code>	companyId, reason
<code>STAFF_CREATED</code>	userId, staffRole

VEHICLE_STATUS_CHANGED

vehicleId, fromStatus, toStatus

16.3 Transaction Safety

All financial and state-mutation operations use:

- `transactionWithTimeout(fn, 10000)` — 10-second timeout
- `SELECT FOR UPDATE NOWAIT` — row-level locking (fails immediately on conflict rather than waiting)
- Automatic rollback on any exception

16.4 Error Response Consistency

All endpoints return:

- Generic error messages (no SQL errors, stack traces, or file paths)
- Consistent JSON structure: `{ "error": "message" }`
- Appropriate HTTP status codes (400, 401, 403, 404, 409, 429, 500)

End of Document 5.4.4 - API Endpoints and Functionality





D O C U M E N T

5.4.5

Authentication Mechanism

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Authentication Mechanism

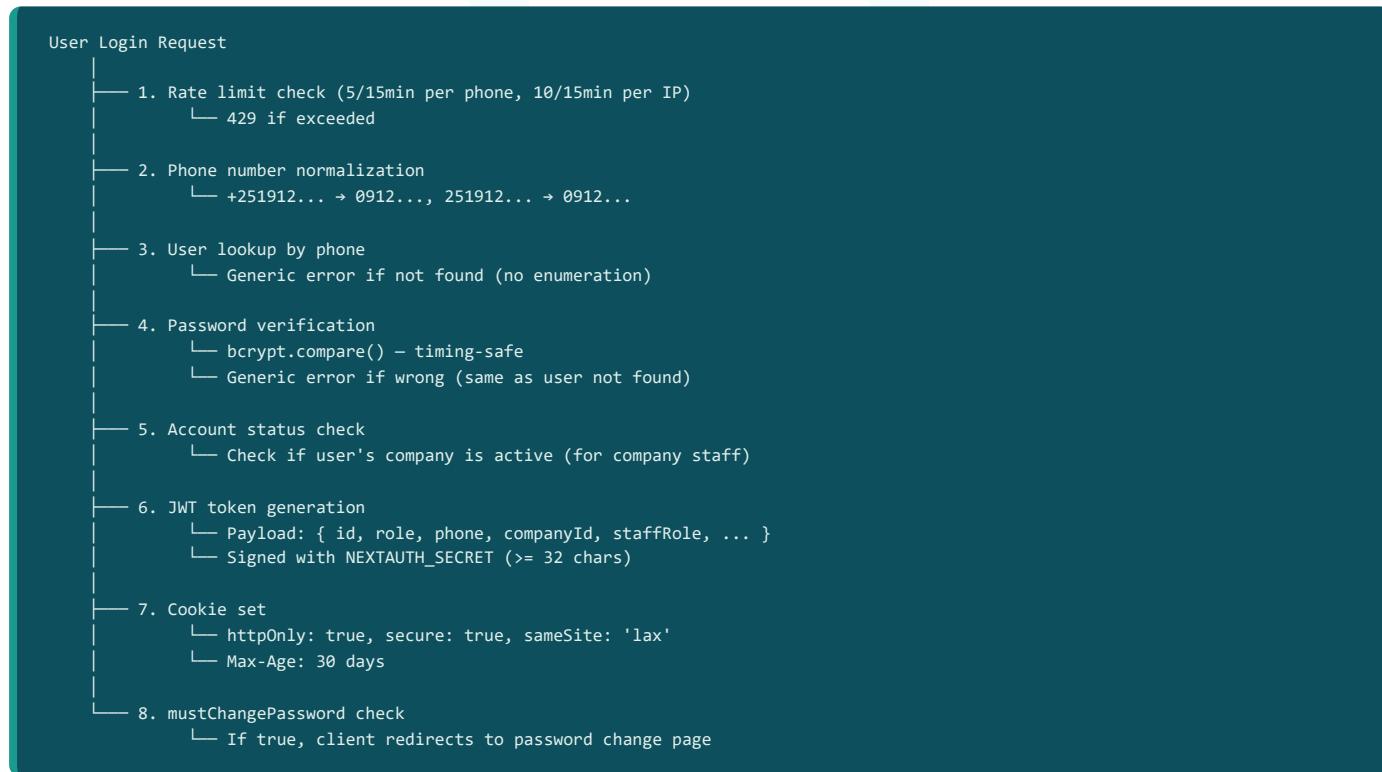
Document: 5.4.5 - Authentication Mechanism **Prepared for:** INSA Cyber Security Audit Division **Application:** i-Ticket Online Bus Ticketing Platform
URL: <https://i-ticket.et> **Version:** v2.14.0 **Date:** February 2026

1. Authentication Architecture

1.1 Primary Authentication

Property	Value
Framework	NextAuth.js v4 (open-source authentication library for Next.js)
Strategy	JWT (JSON Web Token) — stateless, cookie-based
Provider	Credentials provider (phone + password)
Password Hashing	bcrypt with 12 salt rounds
Session Duration	30 days
Cookie Transport	HTTP-only secure cookie

1.2 Authentication Flow



2. Session Management

2.1 JWT Session Configuration

Property	Value	Security Rationale



httpOnly	true	Prevents JavaScript access to cookie (XSS protection)
secure	true (production)	Cookie only sent over HTTPS
sameSite	lax	Prevents cross-origin POST requests from including cookie (CSRF protection)
maxAge	2,592,000 seconds (30 days)	Session duration
path	/	Cookie available to all routes
domain	i-ticket.et	Scoped to production domain

2.2 JWT Token Contents

The JWT payload contains minimal Personally Identifiable Information (PII):

```
{
  "id": "clxxx...",
  "name": "User Name",
  "phone": "0912345678",
  "role": "COMPANY_ADMIN",
  "companyId": "c1yyy...",
  "companyName": "Selam Bus",
  "companyLogo": "/uploads/logo.png",
  "staffRole": "ADMIN",
  "profilePicture": null,
  "nationalId": null,
  "mustChangePassword": false,
  "iat": 1708300000,
  "exp": 1710892000
}
```

2.3 Session Validation Per-Request

Each API request:

1. Extracts JWT from cookie
2. Verifies JWT signature against `NEXTAUTH_SECRET`
3. Checks token expiry
4. Returns session object with user claims

No database lookup per request (stateless JWT). User changes (role updates, deactivation) take effect on next login or token refresh.

2.4 Session Termination

Method	Implementation
User logout	Client-side cookie clearing via NextAuth <code>signOut()</code>
Token expiry	Natural expiry after 30 days
Password change	Does NOT invalidate existing tokens (known limitation)
Account deactivation	Company deactivation blocks login; existing tokens valid until expiry

Known Limitation: JWT has no server-side revocation mechanism. A compromised token remains valid until its 30-day expiry. This is documented in the Known Limitations section of Document 4.2.5.



3. Password Security

3.1 Password Storage

Property	Value
Algorithm	bcrypt
Salt rounds	12
Storage	Hash only (plaintext never stored, never logged)
Comparison	<code>bcryptjs.compare()</code> — constant-time comparison

3.2 Password Requirements

Requirement	Rule
Minimum length	8 characters
Maximum length	72 characters (bcrypt limit)
Uppercase	At least 1 uppercase letter (A-Z)
Lowercase	At least 1 lowercase letter (a-z)
Digit	At least 1 digit (0-9)
Special characters	Not required

Enforced by Zod validation: `.min(8).max(72).regex(/[A-Z]/).regex(/[a-z]/).regex(/[0-9]/)`

3.3 Password Reset Flow

1. User submits phone number
 - └ Rate limit: 3/hr per IP
 - └ Same response regardless of phone existence
2. System generates token
 - └ `crypto.randomBytes(32).toString('hex')` — 256-bit random
 - └ Token hashed with bcrypt before database storage
 - └ Expiry: 1 hour
3. SMS sent to user
 - └ Contains raw token in reset URL
 - └ URL: https://i-ticket.et/reset-password?token=<raw_token>
4. User submits new password + token
 - └ Token compared against stored bcrypt hash
 - └ New password validated (strength rules)
 - └ New password hashed with bcrypt (12 rounds)
 - └ Reset token cleared from database
5. Token is one-time use
 - └ Cleared after successful reset
 - └ Expires after 1 hour if unused

3.4 Temporary Passwords

When a company admin creates staff accounts, a temporary password is generated:

- Staff member must change password on first login (`mustChangePassword: true`)
- System forces redirect to password change page
- Flag cleared after successful password change

4. Role-Based Access Control (RBAC)

4.1 User Roles

Role	Description	Count
CUSTOMER	End users who book tickets	Majority of users
COMPANY_ADMIN	Bus company staff (with sub-roles)	Per-company
SUPER_ADMIN	Platform administrators	Limited (1-3)
SALES_PERSON	Sales agents with referral codes	Variable

4.2 Staff Sub-Roles (under COMPANY_ADMIN)

Sub-Role	Permissions
ADMIN	Full company management
SUPERVISOR	Read-only + limited management (no auto-halt settings)
DRIVER	Trip status updates, GPS tracking
CONDUCTOR	Ticket verification, boarding management
MANUAL_TICKETER (Cashier)	Manual ticket sales, boarding checklist
MECHANIC	Work order management, parts tracking
FINANCE	Cost approval, financial reporting

4.3 Auth Helper Functions

```

requireAuth()      → Any authenticated user
requireRole([...]) → Specific roles only
requireCompanyAdmin() → COMPANY_ADMIN with companyId
requireSuperAdmin() → SUPER_ADMIN only
requireSalesPerson() → SALES_PERSON only
requireFullAdmin() → COMPANY_ADMIN with staffRole null or "ADMIN" (blocks supervisors)
    
```

4.4 Permission Matrix

Resource	Customer	Company Admin	Staff/Cashier	Mechanic	Finance	Super Admin
Book tickets	Yes	No	No	No	No	No
View own bookings	Yes	—	—	—	—	Yes (all)
Manage trips	No	Own company	Own assigned	No	No	All
Manage vehicles	No	Own company	No	No	No	All
Verify tickets	No	Own company	Own company	No	No	All
Sell manual tickets	No	Own company	Own company	No	No	No
Mark no-shows	No	Own company	No	No	No	Yes
Manage work orders	No	Own company	Own assigned	Own assigned	Own company	No
View analytics	No	Own company	No	No	No	All
Manage companies	No	No	No	No	No	Yes
Manage platform staff	No	No	No	No	No	Yes
GPS tracking	No	Own company fleet	Own trip	No	No	All

5. Alternative Authentication Mechanisms

5.1 Cron Job Authentication

Property	Value
Method	Bearer token in <code>Authorization</code> header
Token	<code>CRON_SECRET</code> environment variable
Validation	Exact string match
Used by	<code>/api/cron/cleanup</code> , <code>/api/cron/trip-reminders</code> , <code>/api/cron/predictive-maintenance</code> , <code>/api/cron/telegram-cleanup</code>

5.2 OsmAnd GPS Token Authentication

Property	Value
Method	Query parameter (<code>?token=<value></code>)
Token	<code>crypto.randomBytes(32).toString('hex')</code> — 256-bit random
Storage	<code>Trip.trackingToken</code> field (unique per trip)
Validation	Database lookup against Trip model
Used by	<code>/api/tracking/osmand</code>

5.3 SMS Session Authentication

Property	Value
Method	<code>smsSessionId</code> in request body
Token	System-generated session ID
Expiry	5 minutes
Used by	<code>/api/bookings</code> (SMS booking flow)

5.4 TeleBirr Webhook Authentication

Property	Value
Method	HMAC-SHA256 signature in request body
Key	<code>TELEBIRR_APP_KEY</code> environment variable
Verification	5-gate: hash → replay → HMAC → timestamp → IP whitelist
Comparison	<code>crypto.timingSafeEqual()</code> (prevents timing attacks)
Used by	<code>/api/payments/telebirr/callback</code>

5.5 Telegram Bot Authentication

Property	Value
Method	Secret token header set during webhook registration
Validation	Telegram-provided secret in <code>X-Telegram-Bot-Api-Secret-Token</code> header
Used by	<code>/api/telegram/webhook</code>

5.6 Guest Booking (No Authentication)

Property	Value
Method	Phone number as identity
Flow	First passenger's phone → create/lookup guest User → associate booking
Verification	Phone payment (TeleBirr) serves as verification
Limitations	Guest users have password: null, cannot log in to the portal

6. Brute Force Protection

6.1 Rate Limiting Configuration

Endpoint	Limit	Window	Key	Behavior
Login	5 requests	15 min	Per phone number	Returns 429 with retryAfter
Login	10 requests	15 min	Per IP address	Returns 429 with retryAfter
Registration	3 requests	1 hour	Per IP address	Returns 429
Password reset	3 requests	1 hour	Per IP address	Returns 429

6.2 Rate Limiter Implementation

- Type: In-memory sliding window counter
- Location: `src/lib/rate-limit.ts`
- Key extraction: `x-forwarded-for` header (from Cloudflare) or `x-real-ip` (from Nginx)
- Response: HTTP 429 with `Retry-After` header and JSON body `{ error, retryAfter }`
- Limitation: Resets on server restart (daily at 3 AM via PM2 cron restart)
- Baseline: Nginx provides additional rate limiting (30 req/sec per IP)

6.3 User Enumeration Prevention

Both "user not found" and "wrong password" return the same error message:

```
{ "error": "Invalid phone number or password" }
```

Response timing is consistent due to bcrypt's constant-time comparison.

7. Secret Management

7.1 Environment Variables

Variable	Purpose	Sensitivity
<code>NEXTAUTH_SECRET</code>	JWT signing key	Critical — validated at startup (>= 32 chars, no placeholder detection)
<code>TELEBIRR_APP_KEY</code>	TeleBirr HMAC key	Critical
<code>TELEBIRR_APP_ID</code>	TeleBirr application ID	High
<code>CRON_SECRET</code>	Cron job authentication	High

TELEGRAM_BOT_TOKEN	Telegram Bot API token	High
SMS_API_KEY	SMS gateway authentication	High
DATABASE_URL	PostgreSQL connection string	Critical

7.2 Secret Validation at Startup

The application validates critical secrets on startup:

- `NEXTAUTH_SECRET` : Must be ≥ 32 characters. Rejects known placeholder values (e.g., "change-me", "secret123").
- `DATABASE_URL` : Must be valid PostgreSQL connection string.

7.3 Secret Storage

- Secrets stored as environment variables on the EC2 instance
- Not committed to version control (`.env` in `.gitignore`)
- Loaded by PM2 via `ecosystem.config.js`

8. Security Headers

Authentication-related HTTP headers set by middleware:

Header	Value	Purpose
Strict-Transport-Security	<code>max-age=31536000; includeSubDomains; preload</code>	Force HTTPS (via Cloudflare HSTS)
X-Content-Type-Options	<code>nosniff</code>	Prevent MIME sniffing
X-Frame-Options	<code>DENY</code>	Prevent clickjacking
X-XSS-Protection	<code>1; mode=block</code>	Legacy XSS filter
Referrer-Policy	<code>strict-origin-when-cross-origin</code>	Limit referrer leakage
Permissions-Policy	<code>geolocation=(self), camera=(), microphone=()</code>	Feature restrictions
Cache-Control	<code>no-store</code>	Prevent caching of authenticated pages
Pragma	<code>no-cache</code>	Legacy cache prevention

9. Known Limitations and Recommendations

Limitation	Risk Level	Current Mitigation	Recommendation
No JWT revocation	Medium	Token expires in 30 days	Implement token blacklist or switch to database sessions for sensitive operations
In-memory rate limiter	Low	Nginx baseline + daily restart	Consider Redis-backed rate limiter for persistence
No MFA/2FA	Medium	Phone verification for guests	Add TOTP or SMS OTP for admin accounts
Password change doesn't invalidate tokens	Low	30-day natural expiry	Implement session version counter in JWT
No account lockout after failures	Low	Rate limiting provides equivalent protection	Consider lockout after N failures with admin unlock
<code>sameSite: lax (not strict)</code>	Low	CSP <code>form-action 'self'</code> compensates	<code>strict</code> would break OAuth flows if ever added



End of Document 5.4.5 - Authentication Mechanism





D O C U M E N T

5.4.6

Third Party Integrations

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Third-Party Integrations

Document: 5.4.6 - Third-Party Integrations **Prepared for:** INSA Cyber Security Audit Division **Application:** i-Ticket Online Bus Ticketing Platform **URL:** <https://i-ticket.et> **Version:** v2.14.0 **Date:** February 2026

1. Integration Overview

i-Ticket integrates with the following third-party services:

#	Service	Purpose	Data Flow	Protocol
1	TeleBirr	Mobile money payments	Bidirectional	HTTPS + HMAC-SHA256
2	Telegram Bot API	Booking via Telegram	Bidirectional	HTTPS + Webhook
3	SMS Gateway	SMS notifications & booking	Bidirectional	HTTPS
4	Cloudflare	CDN, WAF, DNS, SSL	Proxy	HTTPS
5	OpenStreetMap Tiles	Map rendering	Outbound (client)	HTTPS
6	OSRM (Project OSRM)	Road route calculation	Outbound (client)	HTTPS
7	Nominatim (OpenStreetMap)	Reverse geocoding	Outbound (client)	HTTPS
8	QR Server API	QR code generation	Outbound (server)	HTTPS
9	Cloudflare Web Analytics	Traffic analytics	Outbound (client)	HTTPS

2. TeleBirr Payment Integration

2.1 Overview

Property	Value
Provider	Ethio Telecom (TeleBirr Super App)
Purpose	Mobile money payment processing
Environment	Production + Demo mode
Transaction Fee	0.5% of transaction amount (absorbed by i-Ticket, invisible to customers)
API Protocol	HTTPS REST API
Authentication	HMAC-SHA256 signed requests

2.2 Data Flow — Payment Initiation

i-Ticket Server → TeleBirr API

Outbound Request:

1. Construct payload: `{ appId, merchantCode, amount, phone, nonce, timestamp, outTradeNo }`
2. Sort parameters alphabetically
3. Sign: `HMAC-SHA256(sortedPayload, TELEBIRR_APP_KEY)`
4. Send POST to TeleBirr API endpoint over HTTPS
5. Include signature in request

Data Sent to TeleBirr:



- App ID (`TELEBIRR_APP_ID`)
- Merchant code
- Payment amount (ETB)
- Customer phone number
- Unique booking reference (`outTradeNo` = booking ID)
- Nonce (replay protection)
- Timestamp
- HMAC-SHA256 signature

2.3 Data Flow — Payment Callback

TeleBirr → i-Ticket Server (POST `/api/payments/telebirr/callback`)

Inbound Webhook — 5-gate verification:

Gate	Check	Failure Response
1	SHA-256 hash of raw payload body	400 — Invalid request
2	Replay detection: lookup hash in <code>ProcessedCallback</code> table	200 — Already processed
3	HMAC-SHA256 verification with <code>crypto.timingSafeEqual()</code>	401 — Invalid signature
4	Timestamp within 5-minute window	400 — Expired callback
5	IP whitelist check (if <code>TELEBIRR_WEBHOOK_IPS</code> configured)	403 — Unauthorized IP

Data Received from TeleBirr:

- Transaction ID
- Booking reference (`outTradeNo`)
- Payment status (SUCCESS / FAILED)
- Amount
- Currency (ETB)
- Timestamp
- HMAC-SHA256 signature

2.4 Environment Variables

Variable	Purpose	Sensitivity
<code>TELEBIRR_APP_ID</code>	Application identifier	High
<code>TELEBIRR_APP_KEY</code>	HMAC signing key	Critical
<code>TELEBIRR_MERCHANT_CODE</code>	Merchant identifier	Medium
<code>TELEBIRR_API_URL</code>	API base URL	Low
<code>TELEBIRR_WEBHOOK_IPS</code>	Allowed callback source IPs (optional)	Medium
<code>DEMO_MODE</code>	If <code>true</code> , bypasses real TeleBirr API	Low

2.5 Security Controls

- HMAC-SHA256 signature on all outbound requests
- 5-gate inbound callback verification (hash, replay, HMAC, timestamp, IP)
- `crypto.timingSafeEqual()` for signature comparison (prevents timing attacks)
- Replay protection via `ProcessedCallback` table
- Server-side amount recalculation (never trusts client-provided amounts)
- Transaction-safe database updates with row locking

3. Telegram Bot Integration

3.1 Overview

Property	Value
Bot Username	@i_ticket_busBot
Bot URL	https://t.me/i_ticket_busBot
Purpose	Bilingual (English/Amharic) ticket booking via Telegram
API	Telegram Bot API v7
Webhook URL	https://i-ticket.et/api/telegram/webhook

3.2 Data Flow

Telegram User → Telegram Servers → i-Ticket Webhook (POST /api/telegram/webhook)

Inbound (from Telegram):

- User messages (text commands)
- Callback query data (button clicks)
- User profile: chat ID, first name, last name, username

Outbound (to Telegram):

- Text messages (booking confirmation, ticket details)
- Inline keyboards (city selection, seat selection, payment options)
- QR code images (ticket QR codes)
- Location data (bus GPS position for /whereismybus)

3.3 Data Stored

`TelegramSession` model:

- `chatId` (BigInt) — Telegram chat identifier
- `phone` — Linked Ethiopian phone number
- `userId` — Linked i-Ticket user ID
- `language` — EN or AM
- `state` — Current conversation state (IDLE, AWAITING_CITY, etc.)
- `data` — JSON blob with temporary booking wizard data
- `expiresAt` — 30-minute session timeout

3.4 Security Controls

- Webhook authenticated via `X-Telegram-Bot-Api-Secret-Token` header
- Session timeout: 30 minutes (cleaned by cron)
- Phone verification links Telegram user to i-Ticket account
- No sensitive data stored in Telegram session (booking data is temporary)
- Rate limited: 100 req/min (WEBHOOK rate limit)

3.5 Environment Variables

Variable	Purpose	Sensitivity
<code>TELEGRAM_BOT_TOKEN</code>	Bot API authentication	Critical



TELEGRAM_BOT_ENABLED	Enable/disable bot	Low
----------------------	--------------------	-----

4. SMS Gateway Integration

4.1 Overview

Property	Value
Purpose	SMS notifications (booking confirmation, ticket delivery, reminders)
Direction	Primarily outbound; inbound for SMS booking flow
Endpoints	<code>/api/sms/incoming</code> (inbound), <code>/api/sms/outgoing</code> (outbound)

4.2 Data Flow

Outbound SMS (i-Ticket → SMS Gateway → Customer):

- Booking confirmation with ticket short codes
- Payment confirmation
- Trip departure reminders
- Password reset tokens
- No-show notifications

Inbound SMS (Customer → SMS Gateway → i-Ticket):

- SMS-based booking flow (USSD-style interaction)

4.3 Data Transmitted via SMS

Data	Example
Ticket short code	ABC123
Trip route	Addis Ababa → Hawassa
Departure time	Feb 20, 2026 at 6:00 AM
Seat number	Seat 12
Tracking URL	https://i-ticket.et/track/ABC123

Not transmitted: Passwords, national IDs, financial details, full booking amounts.

4.4 Security Controls

- HMAC-SHA256 signature on inbound webhooks
- Rate limited
- SMS content limited to non-sensitive operational data
- No PII beyond phone number and passenger name in SMS

4.5 Environment Variables

Variable	Purpose	Sensitivity
<code>SMS_API_KEY</code>	Gateway authentication	Critical
<code>SMS_API_URL</code>	Gateway endpoint	Low
<code>SMS_WEBHOOK_SECRET</code>	Inbound webhook verification	High

5. Cloudflare Integration

5.1 Overview

Property	Value
Purpose	CDN, WAF, DDoS protection, DNS management, SSL termination
Domain	i-ticket.et
SSL Mode	Full (Strict)
Plan	Free tier

5.2 Services Used

Service	Configuration
DNS	Cloudflare nameservers manage i-ticket.et
SSL/TLS	Full (Strict) mode — origin has valid SSL certificate
TLS Minimum	TLS 1.2 (TLS 1.0 and 1.1 disabled)
HSTS	Enabled with preload, includeSubDomains, max-age 1 year
CDN	Caches static assets (JS, CSS, images, fonts)
DDoS	Automatic DDoS mitigation
WAF	Managed ruleset (free tier)
Bot Protection	Basic bot protection
Web Analytics	Cloudflare Web Analytics beacon

5.3 Disabled Features

Feature	Reason
Email Obfuscation	Modifies HTML, breaks React hydration (documented bug)
Rocket Loader	Interferes with Next.js JavaScript loading
Auto Minify	Next.js handles its own minification

5.4 Data Flow

User Browser → Cloudflare Edge → Nginx (54.147.33.168) → Next.js Application

Cloudflare acts as a reverse proxy. All traffic passes through Cloudflare:

- Client IP forwarded via `CF-Connecting-IP` and `X-Forwarded-For` headers
- Cloudflare terminates SSL, re-encrypts to origin (Full Strict)
- Static assets cached at Cloudflare edge (global CDN)
- Dynamic API requests proxied to origin

5.5 Security Headers (via Cloudflare)

- `Strict-Transport-Security: max-age=31536000; includeSubDomains; preload`
- TLS 1.2 minimum enforced at edge
- Automatic HTTPS redirect

6. OpenStreetMap Tile Server

6.1 Overview

Property	Value
Purpose	Map tile rendering for GPS tracking, fleet maps, pickup/dropoff selection
Provider	OpenStreetMap Foundation (open source)
URL	https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png
Integration	Client-side only (Leaflet.js map library)

6.2 Data Flow

User Browser → OpenStreetMap CDN (tile images)

- Map tiles are fetched directly by the user's browser
- No server-side requests to OpenStreetMap
- No user data sent to OpenStreetMap (only tile coordinates)
- Tiles cached by browser and service worker

6.3 CSP Configuration

```
img-src 'self' https://*.tile.openstreetmap.org;
connect-src 'self' https://*.tile.openstreetmap.org;
```

7. OSRM (Open Source Routing Machine)

7.1 Overview

Property	Value
Purpose	Road route geometry for tracking map overlays
Provider	Project OSRM (open source, public demo server)
URL	https://router.project-osrm.org/route/v1/driving/{lon1},{lat1};{lon2},{lat2}
Integration	Client-side only (fetched once per map load)

7.2 Data Flow

User Browser → OSRM Public API → Returns road geometry (GeoJSON)

- Only sends origin/destination coordinates (no user data)
- Route geometry rendered as blue line on tracking maps
- Fetched once per component mount (not on polling intervals)
- Falls back to straight dashed line if OSRM unreachable

7.3 CSP Configuration

```
connect-src 'self' https://router.project-osrm.org;
```

8. Nominatim (OpenStreetMap Geocoding)

8.1 Overview

Property	Value
Purpose	Reverse geocoding for pickup/dropoff location names
Provider	OpenStreetMap Foundation (open source)
URL	https://nominatim.openstreetmap.org/reverse
Integration	Client-side only (on map click for stop selection)

8.2 Data Flow

```
User Browser → Nominatim API (latitude, longitude) → Returns place name
```

- Only sends coordinates when user clicks on map to select pickup/dropoff
- Returns human-readable address/place name
- No user data or session information sent

8.3 CSP Configuration

```
connect-src 'self' https://nominatim.openstreetmap.org;
```

9. QR Server API

9.1 Overview

Property	Value
Purpose	QR code image generation for tickets
Provider	goQR.me (public API)
URL	https://api.qrserver.com/v1/create-qr-code/
Integration	Server-side (QR URLs embedded in ticket data)

9.2 Data Flow

```
i-Ticket Server generates URL → Browser loads QR image from QR Server
```

- QR code URL format: https://api.qrserver.com/v1/create-qr-code/?size=200x200&data=<ticket_code>
- Only ticket short codes (6-char alphanumeric) sent as `data` parameter
- No PII or sensitive data in QR code content
- QR image loaded client-side from the generated URL

9.3 CSP Configuration

```
img-src 'self' https://api.qrserver.com;
```

10. Cloudflare Web Analytics

10.1 Overview

Property	Value
Purpose	Privacy-friendly traffic analytics (no cookies)
Provider	Cloudflare
Script URL	https://static.cloudflareinsights.com/beacon.min.js
Beacon URL	https://cloudflareinsights.com
Integration	Client-side JavaScript beacon

10.2 Data Collected

- Page views and navigation
- Browser/device information
- Geographic location (country level)
- Referrer URLs

Not collected: Personal data, cookies, IP addresses (privacy-first analytics).

10.3 CSP Configuration

```
script-src 'self' https://static.cloudflareinsights.com;
connect-src 'self' https://cloudflareinsights.com;
```

11. Integration Security Summary

Integration	Auth Method	Data Sensitivity	Direction	Encryption
TeleBirr	HMAC-SHA256	High (financial)	Bidirectional	TLS 1.2+
Telegram	Bot token + secret header	Medium (booking data)	Bidirectional	TLS 1.2+
SMS Gateway	API key + HMAC	Medium (ticket codes, names)	Bidirectional	TLS 1.2+
Cloudflare	Origin certificate	Low (proxied traffic)	Proxy	TLS 1.2+ (Full Strict)
OpenStreetMap	None (public)	None (tile coordinates only)	Client outbound	TLS
OSRM	None (public)	None (coordinates only)	Client outbound	TLS
Nominatim	None (public)	None (coordinates only)	Client outbound	TLS
QR Server	None (public)	Low (ticket short codes)	Client outbound	TLS
Cloudflare Analytics	None (injected script)	Low (page views only)	Client outbound	TLS

12. Data Shared with Third Parties

Third Party	Data Shared	Data NOT Shared
TeleBirr	Phone number, payment amount, booking reference	Passenger names, national IDs, trip details
Telegram	Chat ID, phone (user-provided), booking details during wizard	Passwords, national IDs, payment details
SMS Gateway	Phone number, passenger name, ticket code, route, departure time	National ID, payment amounts, financial data

Cloudflare	All proxied HTTP traffic (encrypted)	Database contents (localhost only)
OpenStreetMap	Map tile coordinates (from browser)	No user data
OSRM	Route coordinates (from browser)	No user data
Nominatim	Clicked map coordinates (from browser)	No user data
QR Server	6-character ticket short code	No PII, no booking details
Cloudflare Analytics	Page view data (no PII)	No personal data

End of Document 5.4.6 - Third-Party Integrations





DOCUMENT

5.4.7

Compliance and Regulatory Requirements

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Compliance and Regulatory Requirements

Document: 5.4.7 - Compliance and Regulatory Requirements Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Applicable Regulatory Frameworks

1.1 Ethiopian Regulations

Regulation	Applicability	Compliance Status
INSA Cybersecurity Framework	Web application security requirements for .et domains	Under audit (this submission)
Ethiopian Data Protection Proclamation	Personal data processing requirements	Compliant (see Section 2)
National Bank of Ethiopia (NBE) Directives	Payment service provider regulations	Compliant via TeleBirr (licensed PSP)
Ethiopian Communications Authority (ECA)	Telecommunications and SMS regulations	Compliant via licensed SMS gateway
Ethiopian Revenue and Customs Authority (ERCA)	VAT collection and reporting	Compliant (15% VAT on commission)
Federal Transport Authority	Bus transport regulations	Bus companies hold individual licenses

1.2 International Standards (Referenced)

Standard	Relevance	Implementation Level
OWASP Top 10 (2021)	Web application security baseline	Fully addressed (see Section 3)
OWASP Secure Coding Practices	Development guidelines	Adopted as coding standard
ISO/IEC 27001	Information security management	Principles applied (not certified)
PCI DSS	Payment card data security	Not applicable (no card data stored — TeleBirr handles payment processing)
NIST Cybersecurity Framework	Risk management	Risk-based approach adopted

2. Data Protection Compliance

2.1 Personal Data Inventory

Data Category	Fields	Legal Basis	Retention
Identity	Name, phone, email	Service delivery (contract)	Account lifetime
National ID	nationalId	Passenger verification (legal obligation)	Account lifetime
Authentication	Password hash	Account security (contract)	Until password change
Financial	Payment amount, transaction ID	Payment processing (contract)	Permanent (audit trail)
Travel	Trip bookings, tickets	Service delivery (contract)	Permanent (business record)
Location	GPS coordinates (driver tracking)	Service delivery (real-time tracking)	7 days (auto-purged by cron)
Communication	SMS content, Telegram session	Service delivery (notifications)	SMS: not stored; Telegram: 30-min session
Audit	AdminLog entries	Security and compliance	Permanent
Banking	Company bank account, branch	Company settlement (contract)	Account lifetime

2.2 Data Minimization

Principle	Implementation
Collect only necessary data	Guest users: phone only. Registered: name + phone + optional email
Limit data in tokens	JWT contains: id, role, companyId, staffRole (no passwords, no national IDs)
Limit data in responses	Ticket verification hides <code>bookedByPhone</code> . Error responses hide internal details
Limit data in SMS	Short codes + route + time only (no financial details, no national IDs)
GPS data purge	GPS positions auto-deleted after 7 days for completed/cancelled trips
Session cleanup	Telegram sessions: 30-min timeout. SMS sessions: 5-min timeout

2.3 Data Subject Rights

Right	Implementation
Access	Users can view all their bookings, tickets, and profile via the portal
Rectification	Users can update their profile (name, email, phone, national ID)
Deletion	Account deletion available upon request (contact support)
Data portability	Booking history and tickets available for download
Objection to processing	Users can contact support to object to data processing

2.4 Consent

Scenario	Consent Mechanism
Account creation	Explicit registration action (filling form + clicking register)
Guest booking	Implicit consent via phone payment (phone number = identity + consent)
SMS notifications	Consent implied by booking (notifications are part of service delivery)
Telegram bot	User initiates interaction by messaging the bot
GPS tracking	Driver activates tracking manually (explicit opt-in per trip)
Analytics	Cloudflare Web Analytics: no cookies, no PII (no consent required)

3. OWASP Top 10 (2021) Compliance

A01:2021 — Broken Access Control

Control	Implementation
Role-based access control	4 primary roles + 7 staff sub-roles
Company segregation	Every company API filters by <code>companyId</code> from session
Resource ownership	Users can only access their own bookings/tickets
Admin privilege separation	<code>requireFullAdmin()</code> blocks supervisors from sensitive settings
View-only trip states	DEPARTED/COMPLETED/CANCELLED trips are read-only
Function-level access control	Auth helpers enforce roles per route handler

A02:2021 — Cryptographic Failures

Control	Implementation
Password hashing	bcrypt with 12 salt rounds
TLS in transit	TLS 1.2+ enforced via Cloudflare
HMAC signatures	SHA-256 for TeleBirr payment verification
Timing-safe comparison	<code>crypto.timingSafeEqual()</code> for all cryptographic comparisons
Secret management	Environment variables, not in code; validated at startup
Gap	No field-level encryption at rest for nationalId, bankAccount (recommended)
Gap	No database encryption at rest (AWS EBS encryption available but not enabled)

A03:2021 — Injection

Control	Implementation
SQL injection	Prisma ORM exclusively — zero raw SQL queries in codebase
XSS	React 18 auto-escaping + CSP headers + Zod input validation
Command injection	No shell command execution from user input
Path traversal	System-generated filenames, no user-controlled file paths

A04:2021 — Insecure Design

Control	Implementation
Threat modeling	Trust boundaries documented (DFD Level 0/1/2)
Business logic protection	RULES.md with mandatory rules, Bug Registry
Rate limiting	Per-endpoint rate limits for all mutation endpoints
Payment amount verification	Server-side recalculation (never trusts client amounts)
Race condition prevention	<code>SELECT FOR UPDATE NOWAIT</code> + transaction timeouts

A05:2021 — Security Misconfiguration

Control	Implementation
Security headers	CSP, HSTS, X-Frame-Options, X-Content-Type-Options, Referrer-Policy
Error handling	Generic error messages in production (no stack traces, SQL, file paths)
Unnecessary features	Cloudflare Email Obfuscation disabled, Rocket Loader disabled
Default credentials	<code>mustChangePassword</code> flag for staff accounts with temp passwords
Gap	<code>unsafe-inline</code> in CSP script-src (Next.js 14 limitation)

A06:2021 — Vulnerable and Outdated Components

Control	Implementation
Dependency management	npm with <code>package-lock.json</code> for deterministic installs
Vulnerability scanning	GitHub Dependabot automated alerts
Runtime versions	Node.js 20 LTS, PostgreSQL 16, Ubuntu 22.04 LTS — all actively supported
Known overrides	<code>glob: ^10.5.0</code> security patch applied

Resolution status	6 of 7 Dependabot alerts resolved as of v2.13.0
-------------------	---

A07:2021 — Identification and Authentication Failures

Control	Implementation
Brute force protection	Rate limiting: 5 login attempts per 15 min (per phone), 10 per IP
Password strength	Minimum 8 chars, uppercase + lowercase + digit required
Credential storage	bcrypt (12 rounds), plaintext never stored
Session management	httpOnly, secure, sameSite cookies; 30-day expiry
User enumeration	Same error for wrong phone and wrong password
Gap	No multi-factor authentication (MFA/2FA)
Gap	No JWT revocation mechanism

A08:2021 — Software and Data Integrity Failures

Control	Implementation
Dependency integrity	<code>package-lock.json</code> lockfile, <code>npm ci</code> for production
Webhook verification	HMAC-SHA256 for TeleBirr callbacks, Telegram secret header
Payment integrity	Server-side amount recalculation, replay protection
Code integrity	Git version control, deployment from specific branch

A09:2021 — Security Logging and Monitoring Failures

Control	Implementation
Audit logging	AdminLog table records all sensitive operations
Application logs	PM2 process manager with log rotation
CSP violation reports	<code>/api/csp-report</code> endpoint, logged to PM2
Cloudflare analytics	Traffic monitoring, DDoS alerting
Gap	No centralized SIEM integration
Gap	No automated alert system for security events

A10:2021 — Server-Side Request Forgery (SSRF)

Control	Implementation
No user-controlled URLs	All external API URLs are hardcoded (TeleBirr, SMS, Telegram)
No URL parameters	No endpoints accept URLs as input parameters
Internal services	Database on localhost (not network-exposed)

4. Financial Compliance

4.1 VAT Compliance

Property	Value
VAT Rate	15% (Ethiopian standard rate)

Applied to	Platform commission (5% of ticket price)
Calculation	<code>commission = ticketPrice * 0.05; vat = commission * 0.15; total = commission + vat</code>
Example	850 ETB ticket → 42.50 commission + 6.38 VAT = 48.88 ETB total charge
Reporting	Tax reports available via /api/admin/tax-reports

4.2 Payment Processing

Property	Value
Payment processor	TeleBirr (Ethio Telecom — licensed PSP)
Card data storage	None (i-Ticket never handles card/bank details)
PCI DSS scope	Out of scope (payment handled entirely by TeleBirr)
Transaction records	All payments logged with transaction IDs, amounts, statuses
Refund policy	Managed by bus companies (not platform-level)
Settlement	Company receives ticket price minus platform commission

4.3 Commission Transparency

Item	Visibility
Platform commission (5%)	Shown to customer as breakdown before payment
VAT on commission (15%)	Shown to customer as breakdown before payment
TeleBirr fee (0.5%)	Absorbed by i-Ticket — invisible to customer
Total customer price	<code>ticketPrice + commission + VAT</code>
Company settlement	<code>ticketPrice</code> (full amount minus platform fee)

5. Transport Sector Compliance

5.1 Passenger Manifest Requirements

Requirement	Implementation
Passenger list	Auto-generated manifests on departure or full capacity
Passenger identification	Name, phone (national ID collected but not in manifest)
Boarding verification	QR code ticket scanning, boarding status tracking
Download format	PDF and Excel manifests available for company admins
Retention	Manifests retained permanently (linked to Trip records)

5.2 Vehicle Safety

Requirement	Implementation
Vehicle inspection	Pre-trip, post-trip, daily, weekly, annual, safety inspections
Risk scoring	AI-based risk scoring (0-100) for each vehicle
Pre-departure check	High-risk vehicles (score >= 85) require recent inspection before departure
Insurance tracking	Insurance expiry date tracked per vehicle
Registration tracking	Registration expiry date tracked per vehicle

Compliance calendar

Upcoming due dates for inspections, renewals

6. Telecommunications Compliance

6.1 SMS Communications

Requirement	Implementation
Licensed gateway	SMS sent via licensed Ethiopian SMS gateway provider
Content restrictions	Operational messages only (booking, payment, reminders)
Opt-out	Contact support to stop SMS notifications
No marketing	No unsolicited marketing SMS sent
Phone format	Ethiopian phone format validated (09/07 prefix)

6.2 Telegram Bot

Requirement	Implementation
Bot registration	Registered with Telegram via @BotFather
User initiation	Users must message the bot first (no unsolicited messages)
Data handling	Minimal data stored (session expires in 30 minutes)
Language support	English and Amharic

7. Privacy Policy and Terms

7.1 Published Policies

Document	URL	Content
Privacy Policy	https://i-ticket.et/privacy	Data collection, usage, retention, rights
Terms of Service	https://i-ticket.et/terms	Service terms, booking conditions, liability
FAQ	https://i-ticket.et/faq	Common questions including privacy, payments

7.2 Key Privacy Disclosures

- Data collected: name, phone, email, booking history, GPS (drivers only)
- Third parties: TeleBirr (payments), SMS gateway (notifications)
- Retention: account data retained until deletion; GPS data purged after 7 days
- Security: encryption in transit (TLS), access controls, audit logging
- Contact: +251 911 550 001

8. Compliance Gaps and Remediation Plan

Gap	Risk	Priority	Recommended Remediation
No field-level encryption for national IDs	Medium	High	Implement AES-256 encryption for <code>nationalID</code> , <code>bankAccount</code> fields
No database encryption at rest	Medium	Medium	Enable AWS EBS encryption on database volume

No MFA for admin accounts	Medium	High	Implement TOTP-based MFA for SUPER_ADMIN and COMPANY_ADMIN roles
No SIEM integration	Low	Medium	Integrate AdminLog with centralized logging service
<code>unsafe-inline</code> in CSP	Low	Medium	Upgrade to Next.js 15+ for nonce-based CSP
JWT has no revocation	Low	Low	Implement token version counter or database session store
No automated security alerts	Low	Medium	Set up alert rules for failed login spikes, unusual API patterns

End of Document 5.4.7 - Compliance and Regulatory Requirements





D O C U M E N T

5.4.8

Authorization and Access Control

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Authorization and Access Control

Document: 5.4.8 - Authorization and Access Control **Prepared for:** INSA Cyber Security Audit Division **Application:** i-Ticket Online Bus Ticketing Platform **URL:** <https://i-ticket.et> **Version:** v2.14.0 **Date:** February 2026

1. Authorization Model

i-Ticket uses Role-Based Access Control (RBAC) with company-level data segregation as a secondary access control layer. Authorization is enforced at the API route handler level (not middleware), ensuring every protected endpoint explicitly checks user role and company ownership.

1.1 Access Control Layers

```

Layer 1: Authentication (Is the user logged in?)
|
Layer 2: Role Check (Does the user have the required role?)
|
Layer 3: Company Segregation (Does the user belong to this company?)
|
Layer 4: Resource Ownership (Does this resource belong to the user?)
|
Layer 5: Status Guards (Is this operation allowed in the current state?)

```

2. User Roles

2.1 Primary Roles

Role	Code	Description	Creation Method
Customer	CUSTOMER	End users who search and book bus tickets	Self-registration or guest creation
Company Admin	COMPANY_ADMIN	Bus company staff members (with sub-roles)	Created by Super Admin or existing Company Admin
Super Admin	SUPER_ADMIN	Platform administrators with global access	Database seeding only
Sales Person	SALES_PERSON	Sales agents who earn referral commissions	Self-registration with referral code

2.2 Company Staff Sub-Roles

When `role = COMPANY_ADMIN`, the `staffRole` field further restricts access:

Sub-Role	Code	Permissions
Admin	ADMIN	Full company management: trips, staff, vehicles, settings, analytics, reports
Supervisor	SUPERVISOR	Read-only dashboard + limited management (cannot change auto-halt settings)
Driver	DRIVER	View assigned trips, update trip status, GPS tracking, field repairs
Conductor	CONDUCTOR	View assigned trips, verify tickets, manage boarding, GPS tracking
Cashier	MANUAL_TICKETER	View assigned trips, sell manual tickets (CASH), boarding checklist
Mechanic	MECHANIC	View/update assigned work orders, track parts, messaging
Finance	FINANCE	Review work order costs, approve expenditures, financial summaries

2.3 Platform Staff Roles (Super Admin sub-roles)

Role	Permissions
Customer Support	View bookings, manage support tickets
Operations Manager	View trips, companies, manage operations
Finance Officer	View revenue, tax reports, settlements
Content Manager	Manage platform content
Compliance Officer	View audit logs, compliance reports

3. Permission Matrix

3.1 API Endpoint Access by Role

API Group	Customer	Company Admin	Driver	Conductor	Cashier	Mechanic	Finance	Super Admin	Sales
Auth (login/register)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Public trips (search)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Own bookings	CRUD	—	—	—	—	—	—	All	—
Create booking	Yes	No	—	—	—	—	—	No	—
Process payment	Yes	—	—	—	—	—	—	—	—
Company trips	—	CRUD	Read/Status	Read	Read	—	—	Read all	—
Manual ticket sale	—	Yes	—	—	Yes	—	—	—	—
Ticket verification	—	Yes	—	Yes	—	—	—	Yes	—
No-show marking	—	Yes	—	—	—	—	—	Yes	—
Replacement tickets	—	Yes	—	—	Yes	—	—	Yes	—
Company staff	—	CRUD	—	—	—	—	—	—	—
Company vehicles	—	CRUD	—	—	—	—	—	—	—
Work orders	—	CRUD	Read/Update	—	—	Read/Update	Read/Update	—	—
Fleet analytics	—	Read	—	—	—	—	—	—	—
Company reports	—	Read/Export	—	—	—	—	—	—	—
Company settings	—	Admin only	—	—	—	—	—	—	—
GPS tracking (send)	—	Yes	Yes	Yes	—	—	—	Yes	—
GPS tracking (view)	Public	Company fleet	Own trip	Own trip	—	—	—	All	—
Admin dashboard	—	—	—	—	—	—	—	Yes	—
Manage companies	—	—	—	—	—	—	—	CRUD	—
Platform analytics	—	—	—	—	—	—	—	Yes	—
Audit logs	—	Own company	—	—	—	—	—	All	—
Sales dashboard	—	—	—	—	—	—	—	—	Yes
Support tickets	Own	—	—	—	—	—	—	All	—
Notifications	Own	Own	Own	Own	Own	Own	Own	Own	Own

3.2 Special Access Rules

Rule	Description
Company Admin cannot book	<code>COMPANY_ADMIN</code> role is explicitly blocked from creating bookings to prevent self-booking fraud
Super Admin cannot book	<code>SUPER_ADMIN</code> role is explicitly blocked from creating bookings
Full Admin guard	<code>requireFullAdmin()</code> blocks <code>SUPERVISOR</code> sub-role from sensitive operations (auto-halt settings)
Guest booking allowed	Unauthenticated users can book via guest flow (phone payment = verification)
Cashier unlimited sales	Cashiers can sell manual tickets even when <code>availableSlots = 0</code> (manual ticketing never blocked)
Replacement sales restricted	Only staff/cashier/admin — not available for online customer booking

4. Company Data Segregation

4.1 Segregation Principle

Every company-scoped database query **MUST** include `companyId` from the authenticated session. This is the platform's most critical security rule (RULE-001).

```
// ENFORCED PATTERN in every company API route:
const data = await prisma.trip.findMany({
  where: {
    companyId: session.user.companyId, // ← MANDATORY filter
    // ... other filters
  }
})
```

4.2 Segregation Scope

Resource	Segregated By	Shared?
Trips	<code>Trip.companyId</code>	No — each company sees only its own trips
Bookings	Via <code>Trip.companyId</code>	No — bookings scoped through trip ownership
Vehicles	<code>Vehicle.companyId</code>	No — each company manages its own fleet
Staff	<code>User.companyId</code>	No — staff visible only within their company
Work Orders	<code>WorkOrder.companyId</code>	No — work orders scoped to company vehicles
Analytics	Filtered by <code>companyId</code>	No — analytics computed per-company
Audit Logs	<code>AdminLog.companyId</code>	No — logs scoped to company (except Super Admin views all)
Cities	No <code>companyId</code>	Yes — shared resource (all companies use same city database)
Bookings (customer view)	<code>Booking.userId</code>	Customer sees own bookings regardless of company

4.3 Super Admin Override

Super Admin (`SUPER_ADMIN`) bypasses company segregation and can:

- View all trips, bookings, manifests across all companies
- Manage any company's settings
- View platform-wide audit logs and analytics
- Access any company's data for platform management

4.4 Cross-Company Isolation Verification

To verify segregation, INSA auditors can:

1. Log in as Selam Bus admin (0922345678)
2. Attempt to access trips from another company via API
3. Verify: only Selam Bus trips are returned
4. Verify: modifying another company's trip ID returns 404 (not found, not 403)

Important: The system returns 404 (not 403) for resources belonging to other companies to prevent information leakage about resource existence.

5. Resource-Level Authorization

5.1 Booking Ownership

Operation	Authorization Rule
View booking	Owner (created the booking) OR company admin (trip's company) OR super admin
Cancel booking	Owner (PENDING only) OR company admin OR super admin
Process payment	Owner only (via session or guest phone match)

5.2 Trip Access

Operation	Authorization Rule
View trip (public)	Anyone (public API)
Create trip	Company admin (own company only)
Edit trip	Company admin (own company) + trip in editable state
Delete trip	Company admin (own company) + no paid bookings
Change status	Company admin (own company) + valid status transition
View boarding	Company admin (own company) + BOARDING/DEPARTED only

5.3 Ticket Verification

Operation	Authorization Rule
Verify ticket	Company admin or super admin + ticket's company matches verifier's company
Mark ticket used	Same as verify + ticket not already used

6. Status-Based Access Control

Beyond role-based checks, many operations are restricted by the current state of the resource:

6.1 Trip Status Guards

Trip Status	Allowed Operations	Blocked Operations
SCHEDULED	Edit, delete, change status, toggle booking	No-show, replacement
DELAYED	Edit, change status, toggle booking	No-show, replacement
BOARDING	Change status, toggle booking, view boarding	Edit, delete, no-show, replacement

DEPARTED	No-show marking, replacement sales, GPS tracking	Edit, delete, online booking
COMPLETED	View only	All mutations
CANCELLED	View only	All mutations
SOLD_OUT (availableSlots=0)	View only, manual ticketing	Online booking

6.2 Booking Status Guards

Booking Status	Allowed Operations	Blocked Operations
PENDING	Cancel, process payment	Ticket generation
PAID	View, verify tickets	Cancel (without refund), re-pay
CANCELLED	View only	All mutations
COMPLETED	View only	All mutations

6.3 Payment Window Guard

- Booking payment must be completed within **10 minutes** of booking creation
- After 10 minutes: payment API returns 400, cron job cancels booking and releases slots
- Enforced server-side: `booking.createdAt + 10min > now`

7. Authorization Enforcement Points

7.1 Server-Side Enforcement (Primary)

All authorization is enforced server-side in API route handlers:

```
// Example: Company trip access
export async function GET(request, { params }) {
  // Layer 1: Authentication
  const session = await requireCompanyAdmin() // Returns 401/403 if fails

  // Layer 3: Company Segregation (in query)
  const trip = await prisma.trip.findFirst({
    where: {
      id: params.tripId,
      companyId: session.user.companyId // ← Company isolation
    }
  })

  // Layer 4: Resource existence (returns 404 for other companies)
  if (!trip) return NextResponse.json({ error: "Trip not found" }, { status: 404 })

  // Layer 5: Status guard
  if (!["BOARDING", "DEPARTED"].includes(trip.status)) {
    return NextResponse.json({ error: "Only available for boarding/departed trips" }, { status: 400 })
  }

  return NextResponse.json({ trip })
}
```

7.2 Client-Side UI Controls (Secondary)

The frontend hides UI elements based on role, but this is for UX only — not a security boundary:

- Navigation menus show only role-appropriate items
- Buttons are hidden/disabled based on trip status
- Forms are disabled for read-only states

All access control is enforced server-side. Client-side hiding provides UX convenience but is not relied upon for security.

8. Audit Trail

8.1 AdminLog Entries

All authorization-relevant operations are logged:

Action	Logged Fields
BOOKING_CREATED	userId, bookingId, tripId, companyId, amount
PAYMENT_PROCESSED	userId, paymentId, bookingId, method, amount
TICKET_VERIFIED	userId, ticketId, tripId, companyId
TICKET_USED	userId, ticketId, passengerId, tripId
TRIP_STATUS_CHANGED	userId, tripId, companyId, fromStatus, toStatus
PASSENGER_NO_SHOW	userId, tripId, passengerIds, noShowCount
REPLACEMENT_TICKET_SALE	userId, tripId, bookingId, seatNumbers, amount
STAFF_CREATED	userId, newUserId, staffRole, companyId
COMPANY_DEACTIVATED	userId, companyId, reason
VEHICLE_STATUS_CHANGED	userId, vehicleId, fromStatus, toStatus
LOGIN_FAILED	phone, IP address, reason

8.2 Log Access

Role	Accessible Logs
Company Admin	Own company's AdminLog entries
Super Admin	All AdminLog entries across all companies
Other roles	No direct audit log access

8.3 Log Export

- Company admins: CSV download of own company logs
- Super admins: CSV download of platform-wide logs
- Logs include: timestamp, userId, action, details (JSON), companyId, IP address

9. Concurrency Controls

9.1 Race Condition Prevention

Operation	Control	Mechanism
Booking creation	Row lock on Trip	<code>SELECT FOR UPDATE NOWAIT</code> — fails immediately on conflict
Payment processing	Row lock on Booking	<code>SELECT FOR UPDATE NOWAIT</code> — prevents double payment
No-show marking	Transaction	<code>transactionWithTimeout(10s)</code> — atomic counter updates
Replacement ticket	Transaction + lock	Row lock + released seat cap check in transaction

Trip status change	Transaction	Status transition validated inside transaction
--------------------	-------------	--

9.2 Transaction Timeouts

All write transactions use `transactionWithTimeout(fn, 10000)` :

- 10-second maximum execution time
- Automatic rollback on timeout
- Prevents deadlocks from holding locks indefinitely
- `NOWAIT` on `SELECT FOR UPDATE` fails immediately rather than waiting for lock

10. Testing Access Control

10.1 Test Accounts Provided

Role	Phone	Password	Company
Super Admin	0911223344	demo123	— (global)
Selam Bus Admin	0922345678	demo123	Selam Bus
Customer	0912345678	demo123	—
Guest	Any 09XXXXXXXX	(none)	—

10.2 Suggested Test Cases

Test	Expected Result
Customer accesses <code>/api/company/trips</code>	401 or 403
Selam Bus admin accesses another company's trip by ID	404 (not 403)
Customer creates booking with tampered amount	Server recalculates; amount ignored
Cashier accesses auto-halt settings	403 (requires full admin)
Guest booking without phone payment	Allowed (phone payment = verification)
Attempt to edit DEPARTED trip	400 (read-only state)
Attempt to book on CANCELLED trip	400 (invalid trip status)
Attempt no-show on SCHEDULED trip	400 (only DEPARTED)
Admin sells replacement when no released seats	400 (released seat cap exceeded)
Expired payment (>10 min)	400 (payment window expired)

End of Document 5.4.8 - Authorization and Access Control



D O C U M E N T

5.4.9

Test Accounts

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Test Accounts

Document: 5.4.9 - Test Accounts Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Web Application Test Accounts

1.1 Login URL

<https://i-ticket.et/login>

1.2 Provided Accounts

#	Role	Phone Number	Password	Company	Access Level
1	Super Admin	0911223344	demo123	— (Global)	Full platform administration
2	Company Admin	0922345678	demo123	Selam Bus	Full company management
3	Customer	0912345678	demo123	—	Ticket booking, own bookings
4	Guest	Any 09XXXXXXXX	(none)	—	Guest booking (no login needed)

1.3 Account Capabilities

Super Admin (0911223344)

- Platform dashboard with global statistics
- Manage all bus companies (create, edit, deactivate)
- View all trips, bookings, manifests across companies
- Platform-wide audit logs
- Revenue analytics and tax reports
- Sales person management and commission payouts
- Platform staff management

Login → Dashboard URL: <https://i-ticket.et/admin/dashboard>

Company Admin — Selam Bus (0922345678)

- Company dashboard with trip and booking stats
- Create and manage trips (scheduling, status changes, manifests)
- Staff management (create drivers, conductors, cashiers, mechanics)
- Vehicle fleet management (add vehicles, inspections, work orders)
- Fleet analytics and predictive maintenance dashboard
- GPS fleet tracking map
- Ticket verification and boarding management
- No-show marking and replacement ticket sales
- Company-level audit logs and reports

Login → Dashboard URL: <https://i-ticket.et/company/dashboard>

Customer (0912345678)

- Search and book bus trips
- Select seats, enter passenger details
- Process payments (Demo mode or TeleBirr)



- View booking history and tickets
- Track buses in real-time (for departed trips)
- Submit support tickets

Login → Home URL: <https://i-ticket.et>

Guest (No Account Required)

- Search trips on home page
- Book tickets without registration
- Enter passenger phone number during booking
- Pay via TeleBirr (phone payment serves as verification)
- Receive tickets via SMS
- Track booking via ticket short code at <https://i-ticket.et/track/<code>>

2. Telegram Bot Test Account

2.1 Bot Access

Property	Value
Bot Username	@i_ticket_busBot
Bot URL	https://t.me/i_ticket_busBot
Commands	/start , /book , /mytickets , /whereismybus , /help , /cancel

2.2 Testing the Bot

1. Open Telegram and search for `@i_ticket_busBot` or visit https://t.me/i_ticket_busBot
2. Send `/start` — bot presents language selection (English / Amharic)
3. Send `/book` — bot starts booking wizard
4. Follow prompts: origin city → destination → date → trip selection → passengers → seats → payment

Note: Bot requires phone verification (links to an i-Ticket user account). Use the Customer phone number `0912345678` when prompted.

3. API Test Access

3.1 Authentication Endpoint

```
POST https://i-ticket.et/api/auth/callback/credentials
Content-Type: application/json

{
  "phone": "0911223344",
  "password": "demo123",
  "redirect": false,
  "callbackUrl": "/"
}
```

The session cookie returned can be used for subsequent API calls.

3.2 Public API Endpoints (No Auth Required)

Endpoint	Description
<code>GET /api/trips?origin=Addis+Ababa&destination=Hawassa</code>	Search trips



GET /api/trips/<tripId>/seats	Seat availability
GET /api/cities	List all cities
GET /api/track/<code>	Look up booking by ticket code
GET /api/tracking/<tripId>	Live bus position (for departed trips)

3.3 Cron Endpoints (Bearer Token Required)

```
GET https://i-ticket.et/api/cron/cleanup  
Authorization: Bearer <CRON_SECRET>
```

Note: `CRON_SECRET` will be provided separately to auditors if cron endpoint testing is required.

4. Payment Testing

4.1 Demo Mode

The platform runs in **Demo Mode** (`DEMO_MODE=true`) which allows payment testing without real TeleBirr transactions:

Property	Value
Payment method	Select "DEMO" during payment
Behavior	Payment instantly succeeds
Tickets generated	Yes (with QR codes and short codes)
SMS sent	Yes (confirmation SMS to passenger phone)
Real money	No real financial transaction

4.2 Testing Payment Flow

1. Log in as Customer (0912345678) or proceed as guest
2. Search for a trip (e.g., Addis Ababa → Hawassa)
3. Select trip, enter passenger details, proceed to payment
4. Choose "Demo" payment method
5. Payment completes instantly
6. Tickets appear with QR codes and short codes

4.3 TeleBirr Live Testing

If live TeleBirr testing is required:

1. Set `DEMO_MODE=false` (coordinated with i-Ticket team)
2. Use a valid TeleBirr-enabled phone number
3. Confirm payment in TeleBirr app
4. Callback webhook processes the result

5. GPS Tracking Testing

5.1 Driver Tracking Page

Property	Value

URL	https://i-ticket.et/driver/track
Access	Log in as a driver/conductor staff member
Prerequisite	Trip must be in DEPARTED status

Testing Steps:

1. Log in as Company Admin (0922345678)
2. Find or create a trip and change its status to DEPARTED
3. The driver tracking page shows live GPS controls
4. Browser requests geolocation permission
5. GPS positions are sent every 10 seconds

5.2 Passenger Live Tracking

Property	Value
URL	<a href="https://i-ticket.et/track/<ticketCode>">https://i-ticket.et/track/<ticketCode>
Access	Public (no login required)
Prerequisite	Trip must be in DEPARTED status with active GPS tracking

5.3 Fleet Map

Property	Value
URL	https://i-ticket.et/company/fleet-tracking
Access	Company Admin (0922345678)
Shows	All company vehicles with active GPS on a single map

6. Key Pages for Security Testing

6.1 Public Pages (No Authentication)

Page	URL	Purpose
Home / Trip Search	https://i-ticket.et	Main entry point
Login	https://i-ticket.et/login	Authentication page
Register	https://i-ticket.et/register	User registration
Booking Tracker	<a href="https://i-ticket.et/track/<code>">https://i-ticket.et/track/<code>	Public booking lookup
Privacy Policy	https://i-ticket.et/privacy	Privacy policy
Terms of Service	https://i-ticket.et/terms	Terms and conditions
FAQ	https://i-ticket.et/faq	Frequently asked questions
Contact	https://i-ticket.et/contact	Contact information

6.2 Authenticated Pages

Page	URL	Required Role
Admin Dashboard	/admin/dashboard	SUPER_ADMIN
Admin Companies	/admin/companies	SUPER_ADMIN

Admin Trips	/admin/trips	SUPER_ADMIN
Admin Bookings	/admin/bookings	SUPER_ADMIN
Admin Manifests	/admin/manifests	SUPER_ADMIN
Company Dashboard	/company/dashboard	COMPANY_ADMIN
Company Trips	/company/trips	COMPANY_ADMIN
Company Staff	/company/staff	COMPANY_ADMIN
Company Vehicles	/company/vehicles	COMPANY_ADMIN
Fleet Analytics	/company/fleet-analytics	COMPANY_ADMIN
Fleet Tracking	/company/fleet-tracking	COMPANY_ADMIN
Company Reports	/company/reports	COMPANY_ADMIN
My Bookings	/bookings	CUSTOMER
My Tickets	/tickets	CUSTOMER
Driver Tracking	/driver/track	DRIVER staff
Cashier Portal	/cashier	CASHIER staff
Mechanic Portal	/mechanic	MECHANIC staff
Finance Portal	/finance	FINANCE staff
Sales Dashboard	/sales/dashboard	SALES_PERSON

7. Environment Information

Property	Value
Production URL	https://i-ticket.et
Server IP	54.147.33.168
SSL	Cloudflare Full (Strict)
TLS Version	1.2 minimum
CDN	Cloudflare
Application Port	3000 (behind Nginx reverse proxy)
Database	PostgreSQL 16.11 (localhost:5432, not externally accessible)

8. Important Notes for Auditors

- Demo Mode:** The platform is in demo mode. Payment processing uses simulated TeleBirr transactions. Switch to live mode requires coordination with i-Ticket team.
- Rate Limiting:** The platform enforces rate limits. If you encounter 429 errors during testing, wait for the rate limit window to reset (times vary by endpoint: 1-60 minutes).
- Database Not Externally Accessible:** PostgreSQL listens on localhost only. No external database connections are possible.
- Cloudflare Proxy:** All traffic passes through Cloudflare. Direct IP access is possible but not recommended for accurate security testing.
- GPS Testing:** GPS tracking features require a trip in DEPARTED status. The company admin can change trip status for testing purposes.

6. **Guest Booking:** To test guest booking flow, use any Ethiopian phone number (09XXXXXXXX format) without logging in. Proceed through the booking flow and select Demo payment.
7. **Session Duration:** Login sessions last 30 days. For security testing, clear cookies between tests to ensure clean session state.

End of Document 5.4.9 - Test Accounts





D O C U M E N T

6

Contact Information

I-TICKET PLATFORM — INSA SECURITY AUDIT DOCUMENTATION

i-Ticket Platform - Contact Information

Document: 6 - Contact Information Prepared for: INSA Cyber Security Audit Division Application: i-Ticket Online Bus Ticketing Platform URL: <https://i-ticket.et> Version: v2.14.0 Date: February 2026

1. Organization Information

Field	Value
Organization Name	(TO BE FILLED)
Business Type	Online Bus Ticketing Platform
Website	https://i-ticket.et
Office Address	(TO BE FILLED)
City	Addis Ababa
Country	Ethiopia

2. Primary Contact (Applicant)

Field	Value
Full Name	(TO BE FILLED)
Title/Position	(TO BE FILLED)
Phone	(TO BE FILLED)
Email	(TO BE FILLED)

3. Technical Contact

Field	Value
Full Name	(TO BE FILLED)
Title/Position	(TO BE FILLED)
Phone	(TO BE FILLED)
Email	(TO BE FILLED)

4. Management Contact

Field	Value
Full Name	(TO BE FILLED)
Title/Position	(TO BE FILLED)
Phone	(TO BE FILLED)
Email	(TO BE FILLED)

5. Security Contact (for audit coordination)

Field	Value
Full Name	(TO BE FILLED)
Title/Position	(TO BE FILLED)
Phone	(TO BE FILLED)
Email	(TO BE FILLED)

6. Platform Support Contact (Public)

Field	Value
Support Phone	+251 911 550 001
Support Email	(TO BE FILLED)
Website	https://i-ticket.et/contact

Note: Please fill in all fields marked (TO BE FILLED) before submitting to INSA.

End of Document 6 - Contact Information