CODEX_HANDOFF_v1
Prime Tech Services – Backend Implementation Specification

STATUS: APPROVED FOR IMPLEMENTATION
VERSION: v1
DATE: 24 January 2026

Table of Contents

1. System Architecture

Backend uses Node.js + TypeScript with Prisma and PostgreSQL. Deployment must be Docker compatible.

Tenancy: schema-only multi-tenant. Every domain table includes **tenantId** and all reads/writes are tenant-scoped.

Identifiers: ORD/INV/RCP/LIC/BKG formats are locked and must be uppercase and pattern-validated.

```
Identifier patterns (regex):
ORD: ^ORD-[0-9]{8}-[A-Z0-9]{4}$
INV: ^INV-[0-9]{4}-[0-9]{6}$
RCP: ^RCP-[0-9]{4}-[0-9]{6}$
LIC: ^LIC-[A-Z0-9]{4}-[A-Z0-9]{4}-[A-Z0-9]{4}$
BKG: ^BKG-[A-Z0-9]{8}$
```

2. API Contracts

Base path is **/api**. JSON only. Auth uses HttpOnly session cookies. Non-2xx must use canonical error contract.

```
Canonical error contract:
{
  "error": {
    "code": "STRING_CODE",
    "message": "Human readable message",
    "details": { "fieldErrors": { "field": ["msg"] }, "meta": {} }
  }
}

Auth:
POST /api/auth/register
POST /api/auth/login
POST /api/auth/logout
GET  /api/auth/me
POST /api/auth/verify-email
POST /api/auth/request-password-reset
POST /api/auth/reset-password

Commerce:
POST /api/checkout/start
GET  /api/orders
GET  /api/orders/{ordId}
GET  /api/invoices/{invNumber}
POST /api/invoices/{invNumber}/pdf
GET  /api/receipts/{rcpNumber}
POST /api/receipts/{rcpNumber}/pdf

Stripe:
POST /api/webhooks/stripe

Licensing:
GET /api/licences
GET /api/licences/{licKey}
POST /api/licences/validate
POST /api/licences/activate

Storage:
POST /api/storage/signed-url
```

```
Downloads:
GET /api/releases
GET /api/account/downloads
POST /api/account/downloads/{releaseId}/signed-url

Booking:
POST /api/bookings
GET /api/bookings/track
GET /api/admin/bookings
PATCH /api/admin/bookings/{bkgRef}
```

3. Database & Prisma Model Summary

All models include tenantId. Use UUID primary keys across tables. Stripe provider IDs are stored as indexed fields (not PKs).

Counters for INV/RCP must be concurrency-safe and tenant+year scoped.

```
Minimum required models:
Tenant, User, Session, EmailVerificationToken, PasswordResetToken,
Order, OrderItem, Invoice, Receipt, Payment, WebhookEvent,
Release, Entitlement, Licence, Activation, Counter, Booking

Counter rules:
- key scope: (tenantId, year, type)
- types: INV, RCP
- format: INV-YYYY-NNNNNN / RCP-YYYY-NNNNNN (6-digit)
- atomic increment in transaction with row lock
```

4. Payments & Stripe (Policy B)

Policy B: Invoice is created at checkout initiation. Receipt is created only after payment confirmation.

Stripe webhook must verify signature and be idempotent via WebhookEvent.

5. Licensing & Activation Rules

Licence key format is LIC-XXXX-XXXX-XXXX (uppercase).

MAX=1: Each licence has at most one active activation. Second activation returns canonical error code **LICENCE_MAX_ACTIVATIONS**.

6. Booking System

Create booking returns BKG-XXXXXXXX.

Track requires bkgRef + email.

Admin status transitions are enforced server-side.

```
Status flow:
NEW -> CONFIRMED -> IN_PROGRESS -> COMPLETED
Optional: CANCELLED (only from NEW or CONFIRMED)
```

7. Storage & Signed URLs

All file access is via signed URLs only (no public S3). Store bucket + objectKey metadata for releases and PDFs.

TTL default is 5–15 minutes.

8. Environment Variables

```
Backend:
DATABASE_URL
NODE_ENV
PORT
SESSION_SECRET
COOKIE_SECURE
COOKIE_SAMESITE
APP_BASE_URL

Stripe:
STRIPE_SECRET_KEY
STRIPE_WEBHOOK_SECRET

S3:
AWS_REGION
AWS_ACCESS_KEY_ID
AWS_SECRET_ACCESS_KEY
S3_BUCKET_PRIVATE

Frontend:
NEXT_PUBLIC_SITE_URL
NEXT_PUBLIC_API_BASE
```

9. Security & Safety Rules

Validate all inputs server-side. Enforce tenant scoping on all reads/writes.

Use HttpOnly session cookies with secure flags.

Role-gate admin endpoints; do not leak privileged data.

Rate limit auth and webhook endpoints (recommended).

10. Execution Rules for Codex

Codex must implement modules in fixed order: Auth/session → checkout start (Order + Invoice) → Stripe webhook + Receipt + Entitlements → Licensing/Activation → S3 signed URLs → Booking → canonical errors → Docker runtime.

Codex must not change contracts or add endpoints not listed.

11. Approved for Implementation

This specification is approved and is the sole source of truth for backend implementation. Any change requires a versioned update and re-approval.