PROJECT: ALEF DELTA ERP - MASTER BLUEPRINT

Client: Alef Delta Trading PLC
Developer: Elevate Digital Solutions (BernOS)
Version: 3.1 (The "Telegram" Edition)
Date: November 2025

1. EXECUTIVE SUMMARY & BUSINESS CONTEXT

Alef Delta supplies manpower (Guards, Cleaners, Drivers) to high-profile clients (Embassies, Banks). The business is labor-intensive, low-margin, and compliance-heavy.

The Problem:

- Ghost workers (payroll fraud).
- No-shows at critical sites (security risk).
- Asset leakage (lost radios/uniforms).
- Manual billing errors.

The Solution:
An ERP system that tightly couples Rostering (Plan) with Attendance (Reality) to automate Payroll and Billing.
Strategic Shift:
We are skipping native mobile apps (APK/IPA) to eliminate installation friction. The field interface will be a Telegram Mini App.

2. FUNCTIONAL REQUIREMENTS

A. HR & Compliance

- **Input:** Mass import of unskilled labor.
- **Critical:** Track expiry of Police Clearance and Probation periods (Proc. 1156/11).
- **Guarantors:** Mandatory capture of Guarantor ID/Photo for every security guard.

B. Operations (The Core Loop)

1. **Roster:** Ops Manager assigns Guard A to Embassy B for Night Shift.
2. **Execution:** Guard A opens the Telegram Bot.
3. **Validation:** Bot opens a Mini App -> GPS checks location -> Clock In allowed.
4. **Exception:** If Guard A is >100m away, Clock In is rejected.

C. Asset Tracking

- Checkout system for Uniforms, Weapons, and Radios.
- If an employee is terminated, system auto-checks for unreturned assets before

processing final pay.

D. Finance (Payroll & Billing)

- **Payroll:** Basic Salary + Overtime - Tax - Pension (7%) - Penalties = Net Pay.
- **Billing:** Generate monthly invoices based on *verified* attendance hours or flat contract rates.

3. TECHNICAL ARCHITECTURE

Stack Recommendations

- **Backend:** Node.js (Express/NestJS) or Python (Django/FastAPI).
- **Database:** PostgreSQL (Required for relational integrity).
- **Frontend (Web HQ):** React.js (Admin Dashboard).
- **Frontend (Field Staff): Telegram Mini App (TMA)**.
  - **Technology:** HTML/JS/React hosted on a secure URL.
  - **Wrapper:** Telegram Bot API.
  - **Auth:** window.Telegram.WebApp.initData validation (No passwords).

4. DATABASE SCHEMA (POSTGRESQL)

Added telegram_chat_id to *users table to support the new auth strategy.*

```
-- ALEF DELTA ERP - MASTER SCHEMA v2.1
-- 0. AUTHENTICATION & RBAC
CREATE TYPE user_role AS ENUM ('SUPER_ADMIN', 'OPS_MANAGER', 'HR_MANAGER',
'FINANCE', 'SITE_SUPERVISOR', 'FIELD_STAFF');

CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    employee_id INT REFERENCES employees(id) ON DELETE SET NULL,
    username VARCHAR(50) UNIQUE NOT NULL,
    password_hash VARCHAR(255), -- Nullable for Telegram users who don't need passwords
    telegram_chat_id BIGINT UNIQUE, -- The new primary key for field staff
    email VARCHAR(100) UNIQUE,
    role user_role NOT NULL DEFAULT 'FIELD_STAFF',
    is_active BOOLEAN DEFAULT TRUE,
    last_login TIMESTAMP,
    created_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE permissions (
    id SERIAL PRIMARY KEY,
    slug VARCHAR(50) UNIQUE NOT NULL,
```

```
    description TEXT
);

CREATE TABLE user_permissions (
    user_id INT REFERENCES users(id),
    permission_id INT REFERENCES permissions(id),
    PRIMARY KEY (user_id, permission_id)
);

-- 1. ORGANIZATION & HR
CREATE TABLE departments (
    id SERIAL PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    description TEXT
);

CREATE TABLE job_roles (
    id SERIAL PRIMARY KEY,
    department_id INT REFERENCES departments(id),
    title VARCHAR(100) NOT NULL,
    base_hourly_rate DECIMAL(10, 2),
    billing_hourly_rate DECIMAL(10, 2),
    requires_license BOOLEAN DEFAULT FALSE
);

CREATE TABLE employees (
    id SERIAL PRIMARY KEY,
    employee_code VARCHAR(20) UNIQUE NOT NULL,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    phone_number VARCHAR(20),
    job_role_id INT REFERENCES job_roles(id),
    status VARCHAR(20) DEFAULT 'ACTIVE',
    hire_date DATE NOT NULL,
    termination_date DATE,
    guarantor_details JSONB,
    police_clearance_expiry DATE,
    bank_account_number VARCHAR(50),
    bank_name VARCHAR(50),
    created_at TIMESTAMP DEFAULT NOW()
);

-- 2. CLIENTS & CONTRACTS
```

```
CREATE TABLE clients (
    id SERIAL PRIMARY KEY,
    company_name VARCHAR(100) NOT NULL,
    tin_number VARCHAR(20),
    contact_person VARCHAR(100),
    contact_phone VARCHAR(20),
    billing_cycle VARCHAR(20) DEFAULT 'MONTHLY',
    address_details JSONB
);

CREATE TABLE client_sites (
    id SERIAL PRIMARY KEY,
    client_id INT REFERENCES clients(id) ON DELETE CASCADE,
    site_name VARCHAR(100) NOT NULL,
    latitude DECIMAL(10, 8),
    longitude DECIMAL(11, 8),
    geo_radius_meters INT DEFAULT 100,
    site_contact_phone VARCHAR(20)
);

CREATE TABLE contracts (
    id SERIAL PRIMARY KEY,
    client_id INT REFERENCES clients(id),
    start_date DATE NOT NULL,
    end_date DATE,
    status VARCHAR(20) DEFAULT 'ACTIVE',
    sla_doc_url TEXT
);

CREATE TABLE contract_services (
    id SERIAL PRIMARY KEY,
    contract_id INT REFERENCES contracts(id),
    job_role_id INT REFERENCES job_roles(id),
    quantity INT NOT NULL,
    rate_per_unit DECIMAL(10, 2) NOT NULL,
    billing_type VARCHAR(20) DEFAULT 'HOURLY'
);

-- 3. OPERATIONS
CREATE TABLE shift_schedules (
    id SERIAL PRIMARY KEY,
    employee_id INT REFERENCES employees(id),
    site_id INT REFERENCES client_sites(id),
```

```
    shift_start TIMESTAMP NOT NULL,
    shift_end TIMESTAMP NOT NULL,
    is_overtime_shift BOOLEAN DEFAULT FALSE,
    status VARCHAR(20) DEFAULT 'SCHEDULED',
    created_by_user_id INT REFERENCES users(id)
);

CREATE TABLE attendance_logs (
    id SERIAL PRIMARY KEY,
    schedule_id INT REFERENCES shift_schedules(id),
    employee_id INT REFERENCES employees(id),
    clock_in_time TIMESTAMP,
    clock_out_time TIMESTAMP,
    clock_in_lat DECIMAL(10, 8),
    clock_in_long DECIMAL(11, 8),
    is_verified BOOLEAN DEFAULT FALSE,
    verification_method VARCHAR(20),
    flagged_late BOOLEAN DEFAULT FALSE,
    verified_by_user_id INT REFERENCES users(id)
);

-- 4. INCIDENTS
CREATE TABLE operational_reports (
    id SERIAL PRIMARY KEY,
    site_id INT REFERENCES client_sites(id),
    reported_by_employee_id INT REFERENCES employees(id),
    report_type VARCHAR(50),
    description TEXT,
    evidence_media_urls JSONB,
    severity_level VARCHAR(20),
    created_at TIMESTAMP DEFAULT NOW()
);

-- 5. ASSETS
CREATE TABLE assets (
    id SERIAL PRIMARY KEY,
    asset_code VARCHAR(50) UNIQUE,
    name VARCHAR(100),
    category VARCHAR(50),
    condition VARCHAR(20) DEFAULT 'NEW',
    purchase_date DATE,
    value DECIMAL(10, 2)
);
```

```
CREATE TABLE asset_assignments (
    id SERIAL PRIMARY KEY,
    asset_id INT REFERENCES assets(id),
    assigned_to_employee_id INT REFERENCES employees(id),
    assigned_at TIMESTAMP DEFAULT NOW(),
    returned_at TIMESTAMP,
    return_condition VARCHAR(20),
    notes TEXT
);

-- 6. FLEET
CREATE TABLE vehicle_trip_logs (
    id SERIAL PRIMARY KEY,
    vehicle_asset_id INT REFERENCES assets(id),
    driver_employee_id INT REFERENCES employees(id),
    start_time TIMESTAMP,
    end_time TIMESTAMP,
    start_km INT,
    end_km INT,
    purpose TEXT,
    fuel_added_liters DECIMAL(5, 2)
);

-- 7. FINANCE
CREATE TABLE payroll_periods (
    id SERIAL PRIMARY KEY,
    start_date DATE NOT NULL,
    end_date DATE NOT NULL,
    processed_date TIMESTAMP,
    status VARCHAR(20) DEFAULT 'DRAFT'
);

CREATE TABLE payslips (
    id SERIAL PRIMARY KEY,
    payroll_period_id INT REFERENCES payroll_periods(id),
    employee_id INT REFERENCES employees(id),
    basic_salary DECIMAL(10, 2) DEFAULT 0,
    total_hours_worked DECIMAL(10, 2),
    overtime_hours DECIMAL(10, 2),
    overtime_amount DECIMAL(10, 2) DEFAULT 0,
    transport_allowance DECIMAL(10, 2) DEFAULT 0,
    taxable_income DECIMAL(10, 2),
```

```
    income_tax DECIMAL(10, 2),
    pension_7_percent DECIMAL(10, 2),
    cost_sharing DECIMAL(10, 2),
    penalty_deductions DECIMAL(10, 2),
    loan_repayment DECIMAL(10, 2),
    net_pay DECIMAL(10, 2),
    status VARCHAR(20) DEFAULT 'PENDING'
);

CREATE TABLE invoices (
    id SERIAL PRIMARY KEY,
    invoice_number VARCHAR(50) UNIQUE NOT NULL,
    client_id INT REFERENCES clients(id),
    issue_date DATE DEFAULT CURRENT_DATE,
    due_date DATE,
    total_amount DECIMAL(12, 2),
    tax_amount DECIMAL(12, 2),
    status VARCHAR(20) DEFAULT 'UNPAID',
    generated_at TIMESTAMP DEFAULT NOW()
);

CREATE TABLE invoice_items (
    id SERIAL PRIMARY KEY,
    invoice_id INT REFERENCES invoices(id),
    description VARCHAR(255),
    quantity DECIMAL(10, 2),
    unit_price DECIMAL(10, 2),
    total_price DECIMAL(10, 2)
);
```

5. FRONTEND SPECIFICATIONS (UI/UX)

A. Telegram Mini App (Target: Field Staff - Guards/Drivers)

- **Entry Point:** Telegram Bot Menu button ("Open ERP").
- **Authentication:** Zero Login. Use initData to verify Telegram ID against users table.
- **Key Screens:**
    1. **Shift Dashboard:**
        - Big Button: "CLOCK IN" (Disabled if GPS > 100m from site).
        - Display: "You are scheduled at: US Embassy".
    2. **Incident Reporter:**
        - Form embedded in Web View.

    - Use <input type="file" capture="camera"> for evidence.
3. **Panic Mode:**
    - Simple bot command /panic or a red button in the Web App that sends immediate alert to HQ channel.

B. Web Dashboard (Target: HQ Staff - Ops/HR/Finance)

- **Framework:** React.js / Next.js.
- **Key Modules:**
    1. **Command Center (Ops Manager):**
        - **Live Map:** Leaflet/Google Maps integration showing active clock-ins.
        - **Telegram Linker:** UI to generate "Invite Links" for new employees to connect their Telegram account to their Employee ID.
    2. **Roster Builder:**
        - Calendar Drag-and-Drop Interface.
        - Conflict detection (Prevent double-booking Guard A).
    3. **Finance:**
        - **Payroll Runner:** "Process September Payroll" button -> Generates payslip rows.
        - **Invoice Generator:** Select Client -> Select Date Range -> Generate PDF.

6. API SPECIFICATIONS (Core Routes)

Authentication

- POST /api/auth/telegram - Validates initData string from Telegram. Returns JWT.
- POST /api/auth/login - Standard username/password for HQ Staff (Admins).

Operations (Telegram Mini App)

- GET /api/my-roster - Fetch upcoming shifts for logged-in Telegram user.
- POST /api/attendance/clock-in
    - **Body:** { latitude, longitude, selfie_image }
    - **Logic:** Check distance to client_sites. Return 403 if > radius.
- POST /api/attendance/clock-out
- POST /api/incidents - Multipart upload for images.

Admin (Web)

- GET /api/employees - Filterable list.
- POST /api/employees/link-telegram - Map a telegram_chat_id to an employee_id.
- POST /api/roster/bulk-assign - Assign multiple shifts.
- POST /api/finance/generate-payroll - Triggers the calculation engine.

7. IMPLEMENTATION ROADMAP

1. **Week 1:** Database setup & **Telegram Bot registration**.
2. **Week 2:** HR Module & **Telegram Account Linking** flow (How a guard connects their

Telegram to your system).

3. **Week 3:** Rostering & **Web App Geolocation logic**.
4. **Week 4:** Field testing the Telegram Mini App on