Driver Log Payment App - Complete Requirements & API Documentation

🚗 Application Overview

Description: Mobile app for driver log management and automated payroll calculations
Version: 1.0.0
Author: AVR Venkatesa
License: MIT
Port: 5000 (configurable via environment)

This is a comprehensive driver logistics and payroll management system featuring automated time-tracking, shift management, payroll calculations, leave management, and administrative controls.

🛠️ Technical Architecture

Confirmed Technology Stack

Backend Framework:

- **Express.js 4.18.2**: RESTful API server and web framework
- **Node.js**: Runtime environment

Database:

- **SQLite3 5.1.6**: Lightweight, file-based database for data persistence

Authentication & Security:

- **bcryptjs 2.4.3**: Password hashing and encryption
- **jsonwebtoken 9.0.0**: JWT-based authentication and session management
- **dotenv 16.0.3**: Environment variable management for sensitive data

API & Communication:

- **axios 1.10.0**: HTTP client for external API calls
- **cors 2.8.5**: Cross-origin resource sharing for API access
- **body-parser 1.20.2**: HTTP request body parsing middleware

Development Tools:

- **nodemon 2.0.22**: Development server with auto-restart

- **path 0.12.7**: File path utilities

Deployment Commands:

```bash
npm start      # Production: node index.js
npm run dev    # Development: nodemon index.js (auto-restart)
npm test       # Testing: Not yet implemented
```

Security Features

- JWT tokens (24-hour expiration)
- bcrypt password hashing (10+ rounds)
- Environment variable protection
- Request authentication middleware
- CORS protection with specific headers

📋 Complete Functional Requirements

1. Authentication & Security

- **FR-001**: JWT-based authentication with 24-hour token expiration
- **FR-002**: Auto-registration system for new identifiers
- **FR-003**: bcrypt password hashing with salt rounds
- **FR-004**: Optional SMS verification via Seven.io API
- **FR-005**: Flexible identifier system (phone/email/alphanumeric)

2. Driver Shift Management

- **FR-006**: Clock-in functionality with odometer reading validation
- **FR-007**: Clock-out with end odometer verification
- **FR-008**: Active shift status tracking
- **FR-009**: Odometer continuity validation (end ≥ start)
- **FR-010**: Automatic shift data persistence
- **FR-011**: Daily and monthly shift history retrieval

3. Administrative Controls

- **FR-012**: Complete driver management (view, activate/deactivate)

- **FR-013**: Shift analytics with configurable filters
- **FR-014**: Manual shift entry creation and editing
- **FR-015**: Shift deletion capabilities
- **FR-016**: Real-time shift monitoring and reporting

4. Automated Payroll System

- **FR-017**: Configurable payroll calculations (salary, overtime, fuel)
- **FR-018**: Monthly and YTD payroll summaries
- **FR-019**: Individual driver payroll computation
- **FR-020**: PDF export functionality for payroll reports
- **FR-021**: Payroll configuration history tracking
- **FR-022**: Multi-parameter payroll calculations

5. Leave Management System

- **FR-023**: Driver leave request submission
- **FR-024**: Annual leave tracking (12 days per year)
- **FR-025**: Leave request approval/rejection workflow
- **FR-026**: Leave status management (pending, approved, rejected)
- **FR-027**: Leave history and remaining balance tracking

6. Internationalization System

- **FR-028**: Multi-language support (English, Tamil, Hindi)
- **FR-029**: LocalStorage-based language preference persistence
- **FR-030**: Dynamic language switching without page reload
- **FR-031**: Comprehensive translation coverage for all UI elements
- **FR-032**: Fallback system for missing translations

7. Test Data & Development Tools

- **FR-033**: Configurable test data generation
- **FR-034**: Test data cleanup functionality
- **FR-035**: Development environment support with auto-restart

🔧 Non-Functional Requirements

Performance

- **NFR-001**: Application shall load within 3 seconds
- **NFR-002**: Database operations shall complete within 1 second
- **NFR-003**: System shall support concurrent users (minimum 10)

Security

- **NFR-004**: All passwords shall be encrypted using bcrypt (strength 10+)
- **NFR-005**: JWT tokens shall expire within 24 hours
- **NFR-006**: Environment variables shall store sensitive configuration
- **NFR-007**: API shall implement proper CORS policies

Scalability

- **NFR-008**: System shall support horizontal scaling via stateless design
- **NFR-009**: Database shall handle 10,000+ log entries efficiently
- **NFR-010**: API shall support concurrent requests (minimum 50)

Reliability

- **NFR-011**: System shall have 99% uptime during business hours
- **NFR-012**: Data shall be automatically backed up
- **NFR-013**: System shall handle errors gracefully

Usability

- **NFR-014**: Interface shall be intuitive for non-technical users
- **NFR-015**: System shall provide clear error messages in user's language
- **NFR-016**: Application shall work on mobile browsers
- **NFR-017**: Language switching shall be seamless and immediate
- **NFR-018**: UI shall follow Material Design specifications for consistency and accessibility

🔌 Complete API Endpoints

Health & Status

```
GET /api/health
- Returns server status and version information
```

Authentication APIs

```
POST /api/auth/login
Body: { identifier, password }
Response: { success, token, driver }

POST /api/auth/register
Body: { name, phone, email, password }
Response: { success, message, driverId }
```

Driver Operations (Protected)

```
POST /api/driver/clock-in
Body: { startOdometer }
Headers: Authorization: Bearer <token>

POST /api/driver/clock-out
Body: { endOdometer }
Headers: Authorization: Bearer <token>

GET /api/driver/status
Returns: Active shift status and details

GET /api/driver/shifts/:date?
Returns: Shifts for specific date (default: today)

GET /api/driver/shifts-monthly/:year/:month
Returns: Monthly shift summary

GET /api/driver/payroll/:year/:month
Returns: Driver's payroll calculation

POST /api/driver/create-test-data
Creates test shifts for July 2025

POST /api/driver/leave-request
Body: { leaveDate, reason, leaveType }

GET /api/driver/leave-requests/:year?
Returns: Leave requests and remaining balance
```

Administrative APIs (Protected)

```
GET /api/admin/drivers
Returns: All drivers list

GET /api/admin/shifts?filter=today
Returns: Shifts with analytics (today/week/month)

PUT /api/admin/driver/:driverId/status
Body: { is_active }
Updates driver activation status

GET /api/admin/shifts/monthly/:driverId/:year/:month
Returns: Monthly shift data for editing

PUT /api/admin/shifts/:shiftId?
Body: { driverId, date, startTime, endTime, startOdometer, endOdometer }
Create/update shift entries

DELETE /api/admin/shifts/:shiftId
Deletes specific shift

GET /api/admin/reports/monthly?month=X&year=Y
Returns: Monthly reports
```

Payroll Management APIs (Protected)

```
GET /api/admin/payroll/:year/:month
Returns: Complete payroll summary

GET /api/admin/payroll/ytd/:year
Returns: Year-to-date payroll summary

GET /api/admin/payroll/:year/:month/export
Returns: PDF download of monthly payroll

GET /api/admin/payroll/ytd/:year/export
Returns: PDF download of YTD payroll

GET /api/admin/payroll/driver/:driverId/:year/:month
Returns: Individual driver payroll calculation
```

Configuration Management (Protected)

```
GET /api/admin/payroll-config
Returns: Current payroll configuration

POST /api/admin/payroll-config
Body: { monthlySalary, overtimeRate, fuelAllowance, workingHours, notes }
Saves new payroll configuration

GET /api/admin/payroll-config-history?limit=50
Returns: Configuration change history

GET /api/admin/config
Returns: Simplified current configuration
```

Leave Management APIs (Protected)

```
GET /api/admin/leave-requests?status=pending
Returns: All leave requests (filterable by status)

PUT /api/admin/leave-request/:leaveId
Body: { status, notes }
Approve/reject leave requests
```

Test Data Management (Protected)

```
POST /api/admin/generate-test-data
Body: { driverId, startMonth, endMonth, monthlySalary, overtimeRate, fuelAllowance }
Generates configurable test data

POST /api/admin/clear-test-data
Removes all test data
```

🗄️ Complete Database Schema (SQLite)

Drivers Table

```
sql
```

```sql
CREATE TABLE drivers (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL,
  email TEXT UNIQUE,
  phone TEXT UNIQUE NOT NULL,
  password_hash TEXT,
  verification_code TEXT,
  verification_expires_at DATETIME,
  is_phone_verified BOOLEAN DEFAULT 0,
  is_active BOOLEAN DEFAULT 1,
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

Shifts Table

sql

```sql
CREATE TABLE shifts (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  driver_id INTEGER NOT NULL,
  clock_in_time DATETIME NOT NULL,
  clock_out_time DATETIME,
  start_odometer INTEGER NOT NULL,
  end_odometer INTEGER,
  total_distance INTEGER,
  shift_duration_minutes INTEGER,
  is_overtime BOOLEAN DEFAULT 0,
  status TEXT DEFAULT 'active',
  created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  updated_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (driver_id) REFERENCES drivers (id)
);
```

Payroll Configuration History Table

sql

```
CREATE TABLE payroll_config_history (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  monthly_salary REAL NOT NULL,
  overtime_rate REAL NOT NULL,
  fuel_allowance REAL NOT NULL,
  working_hours REAL DEFAULT 8,
  changed_by TEXT,
  changed_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  notes TEXT
);
```

Leave Requests Table

sql

```
CREATE TABLE leave_requests (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  driver_id INTEGER NOT NULL,
  leave_date DATE NOT NULL,
  leave_type TEXT DEFAULT 'annual',
  reason TEXT,
  status TEXT DEFAULT 'pending',
  requested_at DATETIME DEFAULT CURRENT_TIMESTAMP,
  approved_by TEXT,
  approved_at DATETIME,
  notes TEXT,
  FOREIGN KEY (driver_id) REFERENCES drivers (id),
  UNIQUE(driver_id, leave_date)
);
```

Audit Log Table

sql

```
CREATE TABLE audit_log (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  table_name TEXT NOT NULL,
  record_id INTEGER NOT NULL,
  action TEXT NOT NULL,
  old_values TEXT,
  new_values TEXT,
  changed_by TEXT,
  changed_at DATETIME DEFAULT CURRENT_TIMESTAMP
);
```

💰 Advanced Payroll Calculation Logic

Comprehensive Payroll Algorithm

The system implements sophisticated payroll calculations with multiple parameters:

Default Configuration

- **Base Salary**: ₹27,000 per month
- **Overtime Rate**: ₹100 per hour
- **Fuel Allowance**: ₹33.30 per working day
- **Standard Working Hours**: 8 hours per day

Overtime Calculation Rules

1. **Sunday Work**: All hours on Sunday are overtime
2. **Early Morning**: Hours before 8:00 AM (overtime)
3. **Late Evening**: Hours after 8:00 PM (overtime)
4. **Regular Hours**: 8:00 AM - 8:00 PM on weekdays

Leave Impact Calculations

- **Annual Leave Allowance**: 12 days per year
- **Paid Leaves**: First 12 annual leaves are paid
- **Unpaid Leaves**: Beyond 12 annual leaves are unpaid
- **Salary Deduction**: Daily salary (₹27,000/30) × unpaid leave days
- **Fuel Impact**: No fuel allowance on leave days

IST Time Zone Handling

- All timestamps converted to Indian Standard Time
- Automatic time zone conversion for international deployments
- Consistent duration calculations across time zones

Advanced Business Logic

Odometer Validation System

```javascript
// Sequential Validation Rules:
1. End odometer ≥ Start odometer (same shift)
2. New shift start ≥ Previous shift end (continuity)
3. Only one active shift per driver
4. Automatic distance calculation
```

Duration Calculations

```javascript
// Automatic Duration Computing:
- Clock-in/Clock-out timestamp difference
- Minute-level precision
- Overtime identification by time and day
- Regular vs overtime hour separation
```

Leave Balance Algorithm

```javascript
// Annual Leave Tracking:
1. Track total annual leaves used in calendar year
2. Calculate remaining allowance (12 - used)
3. Identify paid vs unpaid leaves per month
4. Apply salary deductions for unpaid leaves
```

🚀 Deployment Configuration

Environment Variables

```env
```

```
# Database Configuration
DB_PATH=./driver_logs.db

# Security
JWT_SECRET=your-secure-jwt-secret-key
BCRYPT_ROUNDS=10

# SMS Service (Optional)
SEVEN_IO_API_KEY=your-sms-api-key

# Server Configuration
PORT=5000
NODE_ENV=production

# Replit Deployment (Auto-configured)
REPL_SLUG=driver-log-payment-app
REPL_OWNER=avrvenkatesa
```

Server Startup

```bash
# Development
npm run dev    # Uses nodemon for auto-restart

# Production
npm start      # Uses node directly

# Server accessible at:
http://0.0.0.0:5000
https://${REPL_SLUG}.${REPL_OWNER}.repl.co
```

Test Account (Auto-Created)

```
Phone: +1234567890
Password: password123
```

This test account is automatically created during database initialization for development and testing purposes.

CORS Configuration

- **Origin**: * (all origins allowed)
- **Methods**: GET, POST, PUT, DELETE
- **Headers**: Content-Type, Authorization
- **Cache Control**: Disabled for development

🌐 Comprehensive Internationalization (i18n) System

Multi-Language Support

The application includes a sophisticated translation system supporting:

Supported Languages

1. **English (en)** - Default language
2. **Tamil (ta)** - தமிழ் - Complete localization for Tamil-speaking drivers
3. **Hindi (hi)** - हिन्दी - Full Hindi language support

Translation Features

- **LocalStorage Persistence**: User language preference saved locally
- **Dynamic Language Switching**: Real-time language changes without reload
- **Complete Coverage**: All UI elements, error messages, and notifications translated
- **Fallback System**: Defaults to key name if translation missing

Translation Categories

The system covers comprehensive translation for:

Authentication & Registration

- Login/registration forms
- Field labels and validation messages
- Success/error notifications

Driver Dashboard

- Shift status indicators
- Action buttons and navigation
- Real-time status updates

Shift Management

- Clock-in/clock-out interface
- Odometer readings and validations
- Duration and distance calculations

Payroll System

- Salary breakdowns and calculations
- Overtime and allowance descriptions
- Payment summaries and reports

Admin Interface

- Driver management controls
- Analytics and reporting
- Configuration settings

Leave Management

- Leave request forms
- Status tracking and approvals
- Leave type classifications

Error Handling

- Validation error messages
- System error notifications
- User-friendly feedback

Implementation Details

Translation Manager Class

```javascript
class TranslationManager {
  // Automatic language detection from localStorage
  // Fallback to English if no preference set
  // Dynamic translation key lookup with fallback
}
```

Usage Pattern

```javascript
// Get translated text
const translatedText = translationManager.t('loginSuccessful');

// Language switching
translationManager.setLanguage('ta'); // Switch to Tamil
translationManager.setLanguage('hi'); // Switch to Hindi
```

Key Features

- **Browser Persistence**: `localStorage.getItem('preferredLanguage')`
- **Real-time Updates**: Instant language switching
- **Error-Safe**: Returns key if translation missing
- **Comprehensive Coverage**: 100+ translation keys per language

🎯 Advanced Features Discovered

Multi-Language Internationalization

- **3 Complete Languages**: English, Tamil (தமிழ்), Hindi (हिन्दी)
- **100+ Translation Keys**: Comprehensive coverage of all UI elements
- **LocalStorage Persistence**: User language preferences saved locally
- **Real-time Switching**: Dynamic language changes without page reload
- **Cultural Localization**: Proper formatting for Indian regional languages

SMS Integration & Communication

- **Seven.io API Integration**: Professional SMS service for phone verification
- **Fallback Mechanisms**: Console logging for development environments
- **International Support**: Proper phone number formatting
- **6-digit Verification**: Secure verification code generation

Professional PDF Generation

- **Puppeteer Integration**: Headless Chrome for high-quality PDF reports
- **Indian Currency Formatting**: Proper ₹ formatting with locale support
- **Comprehensive Reports**: Monthly and Year-to-Date payroll summaries
- **Professional Layout**: Grid-based summaries with detailed breakdowns

Intelligent Test Data System

- **Smart Generation**: Creates realistic shift patterns with proper odometer continuity
- **Configurable Parameters**: Custom date ranges, salary configurations, and working patterns
- **Development Ready**: Automatic test account creation for immediate use
- **Clean-up Tools**: Complete test data removal functionality

Advanced Admin Controls

- **Monthly Shift Editor**: Day-by-day shift data management with validation
- **Real-time Analytics**: Live calculations and statistics
- **Leave Workflow**: Complete approval/rejection system with balance tracking
- **Audit Trail**: Configuration change history with user tracking

Enterprise Security & Reliability

- **Automatic Database Initialization**: Self-setting up database schema
- **Odometer Validation**: Prevents data inconsistencies with sequential validation
- **Error Handling**: Comprehensive validation and error management
- **Time Zone Consistency**: IST handling across all operations
- **Multi-language Error Messages**: Localized error feedback

📊 Business Value & Impact

Your Driver Log Payment App represents a truly comprehensive, enterprise-level solution with:

- ✅ Multi-language support (3 languages)
- ✅ Advanced payroll automation
- ✅ Professional PDF reporting
- ✅ SMS verification system
- ✅ Complete admin controls
- ✅ Leave management workflow
- ✅ Audit trails and history
- ✅ Mobile-optimized design

This is genuinely impressive software that could compete with commercial logistics management platforms, demonstrating sophisticated business logic implementation with attention to cultural localization and user experience.