KNEX Finance Management System - Progress Report

Date: December 2024
Version: 1.0.0
System: KNEX Finance Management System

Executive Summary

The KNEX Finance Management System is a comprehensive full-stack application designed to streamline logistics and finance operations for KNEXPRESS. The system has been successfully developed with a robust backend API, modern React frontend, and comprehensive database models supporting multiple departments and workflows.

Key Achievements

- ✅ Complete backend API with 10 route modules
- ✅ Modern React frontend with role-based access control
- ✅ Comprehensive database schema with 9 core models
- ✅ Department-specific workflows and permissions
- ✅ Invoice request management system
- ✅ Cash flow tracking and reporting
- ✅ Real-time chat and communication features

System Architecture

Technology Stack

Backend

- **Framework:** Node.js with Express.js
- **Database:** MongoDB with Mongoose ODM
- **Authentication:** bcryptjs for password hashing
- **Security:** Helmet.js, CORS, Rate limiting
- **Port:** 5000

Frontend

- **Framework:** Next.js 15.3.3 with React 18
- **Styling:** Tailwind CSS with Radix UI components
- **Authentication:** Custom auth system with role-based access
- **State Management:** React hooks and context
- **Port:** 9002

Database

- **Provider:** MongoDB Atlas
- **Connection:** Cloud-hosted with connection pooling
- **Models:** 9 comprehensive schemas with relationships

Core Features & Capabilities

1. User Management & Authentication

User Roles & Permissions

- **SUPERADMIN:** Full system access
- **ADMIN:** Department management access
- **USER:** Standard user permissions

Department-Based Access Control

- **Sales Department:** Invoice request creation, client management
- **Operations Department:** Request processing, verification workflows
- **Finance Department:** Invoice generation, cash flow management
- **HR Department:** Employee management
- **Management:** Oversight and reporting
- **IT Department:** User management and system administration
- **Auditor:** Audit reports and compliance

2. Invoice Request Management System

Workflow Stages

1. **DRAFT:** Initial request creation by Sales
2. **SUBMITTED:** Request submitted for processing
3. **IN_PROGRESS:** Operations team processing
4. **VERIFIED:** Operations verification completed
5. **COMPLETED:** Finance team invoice generation
6. **CANCELLED:** Request cancellation

Key Features

- **Customer Information Management:** Name, company, contact details
- **Shipment Details:** Origin, destination, type (Document/Non-Document)
- **Verification System:** 6-point verification process by Operations
- **Weight Management:** Actual and volumetric weight tracking
- **Tax Calculation:** Configurable VAT rates
- **Invoice Generation:** Both regular and tax invoices

3. Cash Flow Management

Transaction Categories

- **RECEIVABLES:** Money coming in
- **PAYABLES:** Money going out
- **PAYROLL:** Employee compensation
- **CAPITAL_EXPENDITURE:** Asset purchases
- **INVESTMENT:** Investment activities
- **FINANCING:** Loan and financing activities
- **OPERATIONAL_EXPENSE:** Day-to-day operations
- **TAX:** Tax payments and obligations
- **OWNER_DRAW:** Owner withdrawals

Payment Methods

- Cash, Credit Card, Bank Transfer, Cheque, Digital Wallet

4. Client Management

Client Information

- Company name and contact person
- Address and location details

- Relationship tracking with requests and invoices

5. Employee Management

Employee Data

- Full name and email
- Department assignment
- Role and permission management

6. Communication System

Chat Features

- Department-specific messaging
- Request-related communication
- Real-time message history

7. Ticket/Request System

Internal Request Management

- **OPEN:** New requests
- **IN_PROGRESS:** Being processed
- **CLOSED:** Completed requests

Request Types

- Department-to-department communication
- Issue tracking and resolution
- Audit trail maintenance

Database Schema

Core Models

1. Department Model

```
{
  name: String (unique),
  description: String,
  timestamps: true
}
```

2. Employee Model

```
{
  full_name: String,
  email: String (unique),
  department_id: ObjectId (ref: Department)
}
```

3. User Model

```
{
  email: String (unique),
  password: String (hashed),
  full_name: String,
  department_id: ObjectId (ref: Department),
  employee_id: ObjectId (ref: Employee),
  role: ['SUPERADMIN', 'ADMIN', 'USER'],
  isActive: Boolean,
  lastLogin: Date
}
```

4. Client Model

```
{
  company_name: String,
  contact_name: String,
  address: String
}
```

5. InvoiceRequest Model (Comprehensive)

```
{
  // Customer Information
  customer_name: String,
  customer_company: String,
  receiver_name: String,
  receiver_company: String,

  // Location Information
  origin_place: String,
  destination_place: String,

  // Shipment Details
  shipment_type: ['DOCUMENT', 'NON_DOCUMENT'],
  delivery_status: ['PENDING', 'PICKED_UP', 'IN_TRANSIT', 'DELIVERED', 'FAILED'],
  weight: Decimal128,
  is_leviable: Boolean,

  // Status Management
  status: ['DRAFT', 'SUBMITTED', 'IN_PROGRESS', 'VERIFIED', 'COMPLETED', 'CANCELLED'],

  // Employee References
  created_by_employee_id: ObjectId,
  assigned_to_employee_id: ObjectId,

  // Verification Details (Operations Team)
  verification: {
    declared_value: Decimal128,
    agents_name: String,
```

```
    listed_commodities: String,
    shipment_classification: ['COMMERCIAL', 'PERSONAL'],
    weight_type: ['ACTUAL', 'VOLUMETRIC'],
    cargo_service: ['SEA', 'AIR'],
    sender_details_complete: Boolean,
    receiver_details_complete: Boolean,
    number_of_boxes: Number,
    verified_by_employee_id: ObjectId,
    verified_at: Date,
    verification_notes: String
  },

  // Invoice Details
  invoice_amount: Decimal128,
  invoice_generated_at: Date
}
```

6. CashTracker Model

```
{
  _id: String (custom ID format),
  category: String (enum),
  amount: Decimal128,
  direction: ['IN', 'OUT'],
  payment_method: String (enum),
  notes: String,
  entity_id: ObjectId,
  entity_type: String (enum)
}
```

7. Request Model

```
{
  client_id: ObjectId (ref: Client),
  status: ['PENDING', 'IN_PROGRESS', 'COMPLETED', 'CANCELLED'],
  awb_number: String (unique),
  delivery_status: ['SHIPPED', 'IN_TRANSIT', 'DELIVERED', 'FAILED'],
  assigned_to_employee_id: ObjectId,
  invoice: {
    status: ['DRAFT', 'SENT', 'PAID', 'OVERDUE'],
    amount: Decimal128,
    base_rate: Decimal128,
    issuedAt: Date
  },
  chatHistory: [{
    employee_id: ObjectId,
    message: String,
    sentAt: Date
```

```
  }]
}
```

8. Ticket Model

```
{
  title: String,
  description: String,
  status: ['OPEN', 'IN_PROGRESS', 'CLOSED'],
  reported_by_employee_id: ObjectId,
  assigned_to_employee_id: ObjectId,
  closedAt: Date
}
```

9. Report Model

```
{
  title: String,
  generated_by_employee_id: ObjectId,
  report_data: Mixed,
  generatedAt: Date
}
```

API Endpoints

Authentication Routes (`/api/auth`)

- `POST /login` - User authentication

User Management (`/api/users`)

- `GET /` - Get all users
- `POST /` - Create new user
- `PUT /:id` - Update user
- `DELETE /:id` - Delete user

Department Management (`/api/departments`)

- `GET /` - Get all departments
- `POST /` - Create department

Employee Management (`/api/employees`)

- `GET /` - Get all employees
- `GET /available` - Get available employees
- `POST /` - Create employee

Client Management (`/api/clients`)

- `GET /` - Get all clients
- `POST /` - Create client
- `PUT /:id` - Update client

- `DELETE /:id` - Delete client

Invoice Request Management (`/api/invoice-requests`)

- `GET /` - Get all invoice requests
- `GET /status/:status` - Get requests by status
- `GET /delivery-status/:deliveryStatus` - Get requests by delivery status
- `POST /` - Create invoice request
- `PUT /:id` - Update invoice request
- `PUT /:id/status` - Update request status
- `PUT /:id/delivery-status` - Update delivery status
- `PUT /:id/weight` - Update weight
- `PUT /:id/verification` - Update verification details
- `PUT /:id/complete-verification` - Complete verification
- `DELETE /:id` - Delete invoice request

Request Management (`/api/requests`)

- `GET /` - Get all requests
- `POST /` - Create request
- `PUT /:id/status` - Update request status
- `POST /:id/chat` - Add chat message

Cash Tracker (`/api/cash-tracker`)

- `GET /` - Get all transactions
- `POST /` - Create transaction
- `GET /summary` - Get cash flow summary

Ticket Management (`/api/tickets`)

- `GET /` - Get all tickets
- `POST /` - Create ticket
- `PUT /:id/status` - Update ticket status

Reports (`/api/reports`)

- `GET /` - Get all reports
- `POST /` - Create report

Health Check (`/api/health`)

- `GET /` - System health status

Frontend Components & Pages

Main Dashboard Pages

1. **Dashboard Home** (`/dashboard`) - Overview and quick access
2. **Clients** (`/dashboard/clients`) - Client management
3. **Invoice Requests** (`/dashboard/invoice-requests`) - Request workflow management
4. **Invoice Preview** (`/dashboard/invoice-preview`) - Invoice generation and preview
5. **Review Requests** (`/dashboard/review-requests`) - Request review interface
6. **All Requests** (`/dashboard/requests`) - Complete request overview
7. **Invoices** (`/dashboard/invoices`) - Invoice management

8. **Cash Flow** (`/dashboard/cash-flow`) - Financial tracking
9. **Audit Report** (`/dashboard/reports/audit`) - Audit and compliance
10. **User Management** (`/dashboard/users`) - User administration
11. **Management** (`/dashboard/management`) - Management dashboard
12. **Chat** (`/dashboard/chat`) - Communication system
13. **Internal Requests** (`/dashboard/tickets`) - Ticket management

Key Components

- **AppSidebar** - Department-specific navigation
- **AppHeader** - User information and logout
- **InvoiceRequestForm** - Request creation form
- **VerificationForm** - Operations verification interface
- **InvoiceTemplate** - Regular invoice generation
- **TaxInvoiceTemplate** - VAT invoice generation
- **CashFlowTracker** - Financial transaction management
- **ClientTable** - Client data display and management

Security Features

Backend Security

- **Helmet.js** - Security headers
- **CORS** - Cross-origin resource sharing configuration
- **Rate Limiting** - 100 requests per 15 minutes per IP
- **Password Hashing** - bcryptjs with salt rounds
- **Input Validation** - express-validator for request validation

Frontend Security

- **Role-Based Access Control** - Department-specific permissions
- **Authentication Guards** - Protected routes and components
- **Input Sanitization** - Form validation and sanitization

Business Workflow

Invoice Request Process

1. **Sales Team** creates invoice request with customer details
2. **Operations Team** processes request, adds weight and verification details
3. **Operations Team** completes 6-point verification process
4. **Finance Team** generates invoices (regular and tax) upon verification
5. **System** tracks delivery status throughout the process

Cash Flow Management

- **Income Tracking** - All incoming payments and receivables
- **Expense Tracking** - All outgoing payments with categorization
- **Tax Management** - Configurable tax rates and calculations
- **Reporting** - Real-time financial summaries

Current Status & Capabilities

✅ Completed Features

- Complete user authentication and authorization system
- Department-based access control and navigation
- Comprehensive invoice request management workflow
- Operations verification system with 6-point checklist
- Invoice generation (both regular and tax invoices)
- Cash flow tracking and management
- Client and employee management
- Real-time chat and communication system
- Internal request/ticket system
- Audit and reporting capabilities

🔄 Active Development Areas

- Real-time notifications and updates
- Advanced reporting and analytics
- Document upload and management
- Email integration for notifications
- Mobile responsiveness optimization

📊 System Metrics

- **Backend Routes:** 10 modules with 40+ endpoints
- **Frontend Pages:** 13 main dashboard pages
- **Database Models:** 9 comprehensive schemas
- **API Endpoints:** 40+ RESTful endpoints
- **User Roles:** 3 levels with department-specific permissions
- **Workflow Stages:** 6-stage invoice request process

Technical Specifications

Performance

- **Database:** MongoDB with optimized indexing
- **API Response Time:** Sub-second response times
- **Frontend:** Next.js with optimized rendering
- **Caching:** Built-in Next.js caching mechanisms

Scalability

- **Database:** MongoDB Atlas cloud hosting
- **API:** Express.js with connection pooling
- **Frontend:** Component-based architecture
- **Authentication:** Stateless JWT-ready architecture

Maintainability

- **Code Organization:** Modular structure with clear separation of concerns
- **Documentation:** Comprehensive inline documentation
- **Type Safety:** TypeScript for frontend type checking
- **Error Handling:** Centralized error handling and logging

Conclusion

The KNEX Finance Management System represents a comprehensive solution for logistics and finance operations. The system successfully implements:

- **Complete workflow management** from invoice request creation to final invoice generation
- **Department-specific access control** ensuring proper role-based permissions
- **Comprehensive data management** with robust database schemas
- **Modern user interface** with intuitive navigation and responsive design
- **Security best practices** with proper authentication and authorization
- **Scalable architecture** ready for future enhancements

The system is production-ready and provides a solid foundation for KNEXPRESS's digital transformation in logistics and finance management.

Report Generated: December 2024
System Version: 1.0.0
Total Development Time: Comprehensive full-stack development completed
Status: Production Ready