🛠️ Technology Stack Documentation

Transportation Management System (TMS)

Author: Aryan Marthak

Project: Transportation Management System

Repository: https://github.com/aryan-marthak/Transportation-Management-System

---

📋 Executive Summary

This document provides a comprehensive overview of all technologies, frameworks, libraries, and tools used in the Transportation Management System. This information is intended for IT department review, security assessment, and authority approval.

System Overview

- **Type:** Full-stack web application
- **Architecture:** Client-Server with RESTful API
- **Deployment:** Docker containerization with cloud-ready configuration
- **Database:** MongoDB Atlas (cloud-hosted)
- **Security:** JWT-based authentication with HTTP-only cookies

---

🏗️ Core Architecture

Backend Architecture

Component	Technology	Version	Purpose
Runtime	Node.js	v18+	JavaScript runtime environment
Framework	Express.js	v5.1.0	Web application framework
Database	MongoDB	v6.17.0	NoSQL document database
ODM	Mongoose	v8.16.1	MongoDB object modeling
Authentication	JWT (jsonwebtoken)	v9.0.2	Token-based authentication
Password Security	bcrypt	v6.0.0	Password hashing

Component	Technology	Version	Purpose
Environment	dotenv	v17.0.1	Environment variable management

Frontend Architecture

Component	Technology	Version	Purpose
Framework	React	v19.1.0	UI library
Build Tool	Vite	v7.0.0	Fast build tool and dev server
Routing	React Router DOM	v7.6.3	Client-side routing
State Management	Zustand	v5.0.6	Lightweight state management
Styling	Tailwind CSS	v3.4.17	Utility-first CSS framework
HTTP Client	Axios	v1.10.0	HTTP request library
Icons	Lucide React	v0.525.0	Modern icon library

🔧 Development Tools & Dependencies

Backend Dependencies

Core Dependencies

```json
{
  "express": "^5.1.0",
  "mongoose": "^8.16.1",
  "jsonwebtoken": "^9.0.2",
  "bcrypt": "^6.0.0",
  "bcryptjs": "^3.0.2",
  "dotenv": "^17.0.1",
  "cors": "^2.8.5",
  "cookie-parser": "^1.4.7",
  "cookies": "^0.9.1",
  "mongodb": "^6.17.0"
}
```

Communication & Notifications

```json
{
  "nodemailer": "^7.0.5",
  "twilio": "^5.8.0"
}
```

Automation & Scheduling

```json
{
  "node-cron": "^4.2.0"
}
```

Development Dependencies

```json
{
  "nodemon": "^3.1.10"
}
```

Frontend Dependencies

Core Dependencies

```json
{
  "react": "^19.1.0",
  "react-dom": "^19.1.0",
  "react-router-dom": "^7.6.3",
  "zustand": "^5.0.6",
  "axios": "^1.10.0",
  "lucide-react": "^0.525.0"
}
```

Data Export

```
{
  "exceljs": "^4.4.0"
}
```

Development Dependencies

```
{
  "vite": "^7.0.0",
  "@vitejs/plugin-react": "^4.5.2",
  "tailwindcss": "^3.4.17",
  "autoprefixer": "^10.4.21",
  "postcss": "^8.5.6",
  "eslint": "^9.29.0",
  "@eslint/js": "^9.29.0",
  "eslint-plugin-react-hooks": "^5.2.0",
  "eslint-plugin-react-refresh": "^0.4.20",
  "@types/react": "^19.1.8",
  "@types/react-dom": "^19.1.6"
}
```

Root Project Dependencies

```
{
  "concurrently": "^8.2.2"
}
```

🐳 Containerization & Deployment

Docker Configuration

- **Docker Compose Version:** 3.8
- **Backend Image:** aryanmarthak/tms-backend
- **Frontend Image:** aryanmarthak/tms-frontend
- **Port Configuration:** Dynamic port mapping
- **Environment Variables:** Externalized configuration

Docker Services

1. **Backend Service**
   - Build context: `./backend`
   - Port mapping: `${PORT}:${PORT}`
   - Environment variables: MongoDB, JWT, Gmail, Port
2. **Frontend Service**
   - Build context: `./frontend`
   - Port mapping: `5173:80`
   - Dependencies: Backend service

🔐 Security Technologies

Authentication & Authorization

- **JWT (JSON Web Tokens):** Stateless authentication
- **HTTP-only Cookies:** XSS protection
- **bcrypt Password Hashing:** Secure password storage
- **Role-based Access Control:** Employee/Admin separation

Data Security

- **MongoDB Security:** Connection string authentication
- **Input Validation:** Comprehensive request validation
- **CORS Configuration:** Cross-origin resource sharing control
- **Environment Variables:** Secure credential management

Communication Security

- **HTTPS Support:** Production-ready SSL/TLS
- **Secure Cookie Settings:** Same-site strict policy
- **API Security:** RESTful API with proper HTTP methods

📡 External Services & APIs

Email Service

- **Provider:** Gmail SMTP
- **Library:** Nodemailer v7.0.5
- **Authentication:** App password-based
- **Features:** HTML email templates, attachment support

SMS Service

- **Provider:** Twilio
- **Library:** Twilio SDK v5.8.0
- **Features:** Programmatic SMS sending
- **Authentication:** Account SID and Auth Token

Database Service

- **Provider:** MongoDB Atlas
- **Type:** Cloud-hosted NoSQL database
- **Connection:** MongoDB driver v6.17.0
- **Security:** Connection string authentication

🛠️ Development & Build Tools

Build System

- **Vite:** Modern build tool for React
- **PostCSS:** CSS processing
- **Autoprefixer:** CSS vendor prefixing
- **Tailwind CSS:** Utility-first CSS framework

Code Quality

- **ESLint:** JavaScript/React linting

- **ESLint Plugins:** React hooks and refresh
- **TypeScript Types:** React type definitions

Development Workflow

- **Nodemon:** Backend auto-restart
- **Concurrently:** Parallel development servers
- **Hot Module Replacement:** Frontend development

📊 Data Management

Database Schema

- **Employee Model:** User management and authentication
- **Driver Model:** Driver information and availability
- **Vehicle Model:** Fleet management and status
- **Trip Request Model:** Request lifecycle and assignment

Data Export

- **ExcelJS:** Excel file generation
- **Report Generation:** Automated data export
- **Format Support:** .xlsx files

🔄 Automation & Scheduling

Background Jobs

- **node-cron:** Scheduled task execution
- **Trip Completion:** Automated status updates
- **Resource Management:** Vehicle/driver availability updates

Real-time Features

- **Polling Mechanism:** 2-5 second intervals

- **Status Updates:** Live request tracking
- **Notification System:** Email and SMS alerts

🌐 Network & Communication

API Architecture

- **RESTful Design:** Standard HTTP methods
- **JSON Data Format:** Lightweight data exchange
- **CORS Support:** Cross-origin requests
- **Error Handling:** Comprehensive error responses

Client-Server Communication

- **Axios:** HTTP client library
- **Cookie Management:** Automatic token handling
- **Request/Response Interceptors:** Centralized error handling

---

📱 User Interface Technologies

UI Framework

- **React 19:** Latest React version
- **Functional Components:** Modern React patterns
- **Hooks:** State and lifecycle management
- **Context API:** Component communication

Styling & Design

- **Tailwind CSS:** Utility-first styling
- **Responsive Design:** Mobile-first approach
- **Lucide Icons:** Modern iconography
- **Component Library:** Custom reusable components

User Experience

- **React Router:** Client-side navigation
- **State Management:** Zustand for global state
- **Form Handling:** Controlled components
- **Loading States:** User feedback mechanisms

🔧 Configuration Management

Environment Variables

- **Backend Configuration:**
    - `PORT` : Server port (default: 5002)
    - `MONGODB_URI` : Database connection string
    - `JWT_TOKEN` : JWT secret key
    - `ADMIN_USER` : Gmail account for notifications
    - `ADMIN_PASS` : Gmail app password
    - `TRANSPORT_HEAD_EMAIL` : Transport head email
    - `TWILIO_*` : SMS service credentials
- **Frontend Configuration:**
    - `VITE_API_URL` : Backend API URL
    - `VITE_NODE_ENV` : Environment mode
    - `VITE_APP_NAME` : Application name

Configuration Files

- **package.json:** Dependency management
- **vite.config.js:** Build configuration
- **tailwind.config.js:** CSS framework config
- **eslint.config.js:** Code quality rules
- **docker-compose.yml:** Container orchestration

🚀 Deployment & Infrastructure

Production Deployment

- **Docker Containers:** Isolated application environments
- **Port Mapping:** Configurable service ports
- **Environment Separation:** Development/Production configs
- **Health Checks:** Container monitoring

Scalability Considerations

- **Stateless Backend:** Horizontal scaling ready
- **MongoDB Atlas:** Cloud database scaling
- **Load Balancing:** Docker service distribution
- **Caching:** Future Redis integration ready

🔍 Monitoring & Logging

Application Monitoring

- **Console Logging:** Development debugging
- **Error Tracking:** Comprehensive error handling
- **Performance Monitoring:** Vite build optimization
- **Database Monitoring:** MongoDB Atlas metrics

Security Monitoring

- **Authentication Logs:** Login attempts tracking
- **API Access Logs:** Request/response monitoring
- **Error Logging:** Security incident tracking

📋 Compliance & Standards

Code Standards

- **ESLint Configuration:** JavaScript/React standards
- **Prettier Integration:** Code formatting
- **Git Hooks:** Pre-commit validation
- **TypeScript Ready:** Type safety preparation

Security Standards

- **OWASP Guidelines:** Web application security
- **JWT Best Practices:** Token security
- **Password Security:** bcrypt hashing
- **HTTPS Enforcement:** Production security

🔄 Version Control & CI/CD

Version Control

- **Git:** Source code management
- **GitHub:** Repository hosting
- **Branch Strategy:** Feature-based development
- **Commit Standards:** Conventional commits

Continuous Integration

- **Build Automation:** Docker image building
- **Testing Framework:** Jest integration ready
- **Code Quality:** ESLint automated checks
- **Deployment:** Docker Compose automation

📚 Documentation & Support

Technical Documentation

- **README.md:** Comprehensive setup guide
- **API Documentation:** Endpoint specifications
- **Code Comments:** Inline documentation
- **Architecture Diagrams:** System design docs

User Documentation

- **Installation Guide:** Step-by-step setup

- **User Manuals:** Employee and Admin guides
- **Troubleshooting:** Common issues and solutions
- **FAQ:** Frequently asked questions

🔮 Future Technology Considerations

Potential Upgrades

- **TypeScript Migration:** Enhanced type safety
- **GraphQL Integration:** Advanced API querying
- **Redis Caching:** Performance optimization
- **WebSocket Integration:** Real-time communication
- **Microservices Architecture:** Service decomposition

Scalability Enhancements

- **Load Balancer:** Traffic distribution
- **CDN Integration:** Static asset delivery
- **Database Sharding:** Horizontal scaling
- **Message Queues:** Asynchronous processing

✅ IT Department Checklist

Security Review

- [ ] **JWT token security assessment**
- [ ] **Password hashing implementation review**
- [ ] **CORS configuration validation**
- [ ] **Environment variable security**
- [ ] **API endpoint security audit**
- [ ] **Database connection security**
- [ ] **External service integration review**

Performance Review

- [ ] Database query optimization
- [ ] Frontend bundle size analysis
- [ ] API response time assessment
- [ ] Memory usage evaluation
- [ ] Scalability planning review

Compliance Review

- [ ] Data protection compliance
- [ ] Privacy policy requirements
- [ ] Audit trail implementation
- [ ] Backup and recovery procedures
- [ ] Disaster recovery planning

Infrastructure Review

- [ ] Docker container security
- [ ] Network configuration validation
- [ ] Monitoring and alerting setup
- [ ] Backup strategy assessment
- [ ] Deployment pipeline review

📞 Contact Information

Developer: Aryan Marthak

Email: [email protected]

Repository: https://github.com/aryan-marthak/Transportation-Management-System

---

📄 Document History

Version	Date	Changes	Author

Version	Date	Changes	Author
1.0	Dec 2024	Initial documentation	Aryan Marthak

Note: This document should be reviewed and updated whenever new technologies are added or existing ones are upgraded. All technology decisions should be approved by the IT department before implementation.