# Meridian Takeoff

## System Architecture & Infrastructure

Version 2.0 - Production Deployment

Comprehensive documentation of the application architecture, hosting infrastructure, and component relationships

## Table of Contents

## 1. Executive Summary

Meridian Takeoff is a professional construction takeoff software built with modern web technologies. The application enables construction professionals to upload PDF drawings, perform precise measurements, manage takeoff conditions, and generate comprehensive reports. The system follows a three-tier architecture: **Frontend:** React + TypeScript application hosted on Vercel **Backend:** Express.js API server hosted on Railway **Database:** PostgreSQL database via Supabase This architecture provides scalability, reliability, and separation of concerns while enabling continuous deployment through GitHub integration.

## 2. Hosting Infrastructure

The application is deployed across three platforms, each serving a specific purpose:

| Platform | Service | Purpose | URL/Configuration |
|---|---|---|---|
| Frontend | Vercel | Static site hosting & CDN | mcw-takeoff-tool.vercel.app |
| Backend API | Railway | Node.js server hosting | mcw-takeoff-tool-production-28fb.up.railway.app |
| Database | Supabase | PostgreSQL database & auth | mxjyytwfhmoonkduvybr.supabase.co |
| Version Control | GitHub | Source code repository | github.com/ubuildacademy/mcw-takeoff-tool |

### Vercel Configuration

Vercel serves the frontend React application with automatic deployments from GitHub. API requests are proxied to Railway via rewrite rules:

```
/api/* → https://mcw-takeoff-tool-production-28fb.up.railway.app/api/*
```

### Railway Configuration

Railway hosts the Express.js backend server with automatic deployments. The server runs on port 4000 (configurable via PORT env variable) and includes:

• Express.js API server

• File upload handling (5GB limit)

• Socket.IO for live preview

• CORS configuration for Vercel domains

• Health check endpoints

• Automatic restarts on failure

### Supabase Configuration

Supabase provides PostgreSQL database with Row Level Security (RLS) and authentication. Key features:

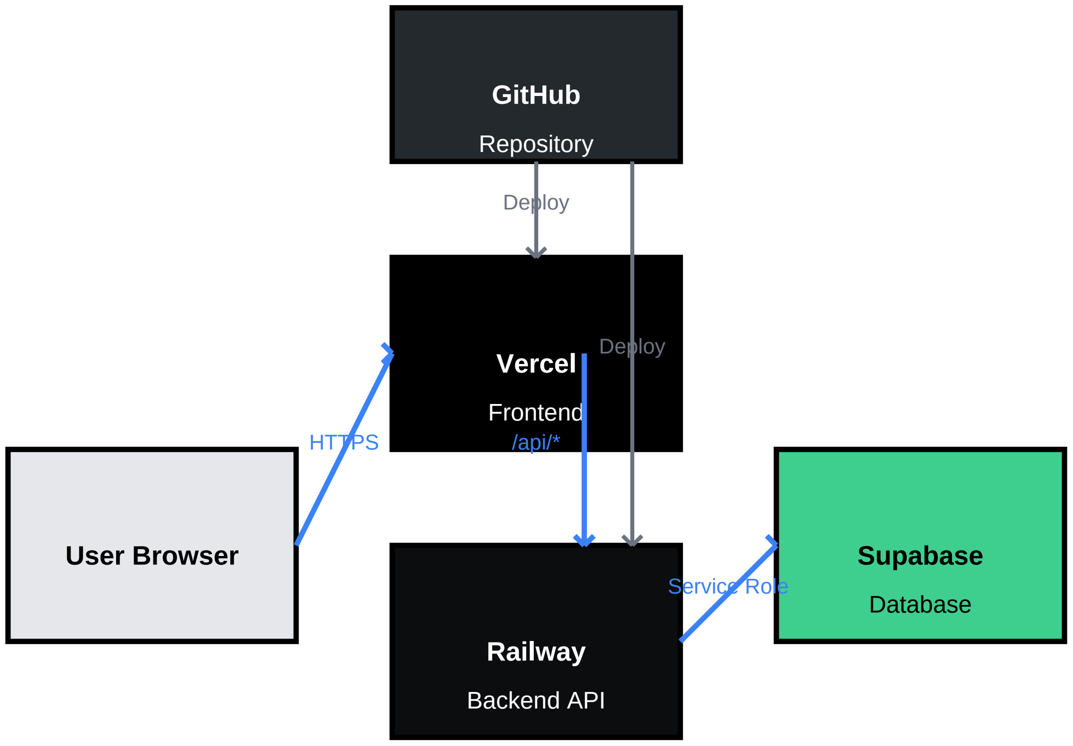
• User authentication & authorization

• PostgreSQL database with real-time subscriptions

- File storage for uploaded PDFs
- Row Level Security policies
- Service role key for backend operations

## 3. System Architecture Diagram

The following diagram illustrates the complete system architecture and data flow:



**Legend:**

• Blue arrows: HTTP/HTTPS requests and API calls

• Gray arrows: Deployment triggers from GitHub

• Black boxes: User-facing components

• Colored boxes: Hosting platforms (Vercel, Railway, Supabase)

## 4. Frontend Architecture

The frontend is a React 18 + TypeScript single-page application built with Vite. It uses modern React patterns including hooks, context, and state management.

### Component Structure

| Component | Purpose | Key Features |
|---|---|---|
| App.tsx | Root router | Route definitions, authentication guards |
| TakeoffWorkspace | Main workspace | PDF viewer, sidebars, tool integration |
| PDFViewer | PDF rendering | PDF.js integration, canvas overlay, zoom/pan |
| TakeoffSidebar | Left sidebar | Conditions, tools, measurement settings |
| SheetSidebar | Right sidebar | Sheet navigation, OCR, labeling |
| ProjectList | Home page | Project dashboard, grid/list views |
| AITakeoffAgent | AI features | AI-powered takeoff automation |
| ChatTab | AI chat | Document-aware chat interface |
| SearchTab | Document search | Full-text search with highlighting |

### State Management

Zustand store (useTakeoffStore) manages global application state:

• Project data and settings

• Current document and page selection

• Takeoff conditions and measurements

• Calibrations and scale settings

• Document annotations and markups

• UI state (sidebar visibility, dialogs, etc.)

### Frontend Services

| Service | Purpose |
|---|---|
| apiService | REST API communication with backend |
| supabaseService | Direct Supabase database operations |
| aiTakeoffService | AI-powered takeoff processing |

| | |
|---|---|
| playwrightTakeoffService | Automated takeoff execution |
| visualSearchService | Visual search functionality |
| ocrService | Client-side OCR (Tesseract.js) |
| backupService | Project backup/restore |

# 5. Backend Architecture

The backend is an Express.js server written in TypeScript, providing RESTful API endpoints and real-time communication via Socket.IO. It handles file processing, AI integrations, and database operations.

## API Routes

| Route | Purpose |
|---|---|
| /api/projects | Project CRUD operations |
| /api/files | File upload and management |
| /api/conditions | Takeoff condition management |
| /api/sheets | Sheet metadata and OCR |
| /api/takeoff-measurements | Measurement storage and retrieval |
| /api/ocr | OCR text extraction |
| /api/enhanced-ocr | Enhanced OCR processing |
| /api/ai-takeoff | AI-powered takeoff processing |
| /api/playwright-takeoff | Automated takeoff execution |
| /api/hybrid-detection | Hybrid detection service |
| /api/visual-search | Visual search functionality |
| /api/ollama | Ollama AI integration |
| /api/users | User management and invitations |
| /api/settings | Application settings |
| /api/calibrations | Scale calibration management |
| /api/rule-validation | Rule-based validation |

## Backend Services

| Service | Purpose |
|---|---|
| aiTakeoffService | Qwen3-VL AI vision processing |
| playwrightTakeoffService | Browser automation for takeoff |
| enhancedOcrService | Enhanced OCR with preprocessing |
| hybridDetectionService | Hybrid AI + rule-based detection |
| visualSearchService | Visual similarity search |
| yoloDetectionService | YOLO object detection |
| qwenVisionService | Qwen Vision API integration |
| livePreviewService | Socket.IO live preview |

| | |
|---|---|
| emailService | Email notifications |
| ruleBasedValidationService | Rule-based validation |

## 6. Database Schema

Supabase PostgreSQL database stores all application data with Row Level Security (RLS) enabled. The database uses snake_case naming conventions and includes comprehensive relationships.

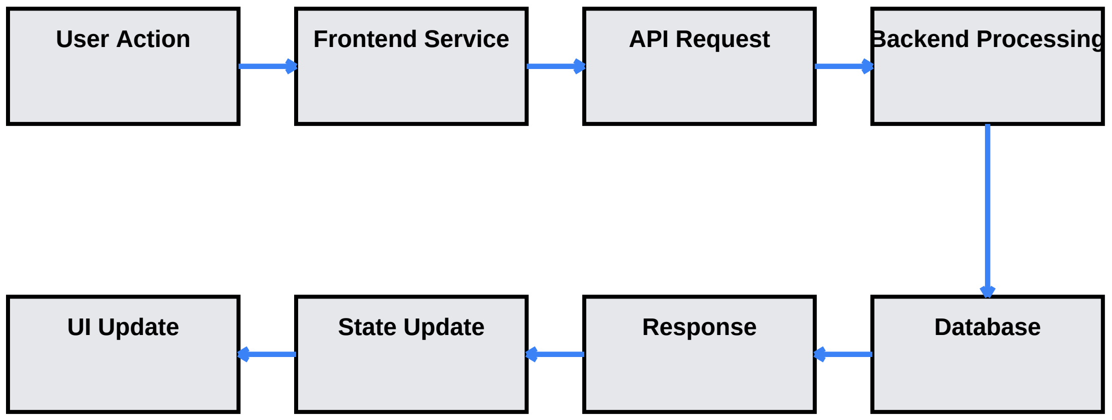
| Table | Purpose | Key Fields |
|---|---|---|
| takeoff_projects | Project metadata | id, name, client, location, status, user_id |
| takeoff_files | Uploaded PDF files | id, project_id, filename, file_path, page_count |
| takeoff_sheets | Sheet metadata | id, document_id, page_number, sheet_name, extracted_text |
| takeoff_conditions | Takeoff conditions | id, project_id, name, type, unit, color, costs |
| takeoff_measurements | Measurements | id, project_id, file_id, pdf_page, points, calculated_value |
| takeoff_calibrations | Scale calibrations | id, project_id, file_id, page_number, scale_factor |
| user_metadata | User profiles | id, role, full_name, company |
| user_invitations | User invitations | id, email, role, invite_token, status |
| app_settings | App configuration | id, key, value |

### Authentication & Authorization

Supabase handles authentication with JWT tokens. The backend uses service role key to bypass RLS for administrative operations, while the frontend uses anon key with RLS policies for data access.

## 7. Data Flow Diagram

The following diagram illustrates the complete data flow from user action to UI update:



The data flow follows this sequence: **User Action:** User interacts with the React frontend (e.g., uploads PDF, creates measurement) **Frontend Service:** Frontend service calls API endpoint via axios or fetch **API Request:** Request is proxied through Vercel to Railway backend (/api/*) **Backend Processing:** Express.js server processes request, validates authentication **Database:** Backend uses Supabase client (service role key) for database operations **Response:** Backend returns JSON response to frontend **State Update:** Frontend updates Zustand store with new data **UI Update:** React components re-render with updated state

### Real-time Features

Socket.IO provides real-time communication for live preview functionality: Frontend connects to Railway server via Socket.IO Backend emits live preview updates during processing Frontend receives updates and updates UI in real-time

## 8. API Routes & Services

See sections 4 and 5 for detailed API routes and services documentation.

# 9. Key Features & Components

Meridian Takeoff provides comprehensive construction takeoff capabilities:

## PDF Processing

• PDF upload and storage

• PDF.js rendering with canvas overlay

• Multi-page navigation

• Zoom, pan, and rotation controls

• OCR text extraction

• Sheet labeling and metadata

## Takeoff Tools

• Area measurements (SF, SY)

• Linear measurements (LF)

• Volume measurements (CF, CY)

• Count measurements

• Scale calibration

• Cutout support for complex shapes

• Perimeter calculations

## Condition Management

• Custom takeoff conditions

• Material and labor cost tracking

• Waste factor management

• Color-coded visualizations

• Unit conversion support

## AI Features

• AI-powered sheet analysis

• Automated takeoff suggestions

• Visual search for similar elements

• Document-aware chat interface

• Hybrid AI + rule-based detection

## Reporting

• Excel export with multiple sheets

• PDF export with visual overlays

• Executive summary reports

• Cost analysis and breakdowns

• Project backup/restore

## Collaboration

• Multi-user support

• Role-based access control

• User invitations

• Project sharing

## 10. Technology Stack

| Category | Technology | Purpose |
|---|---|---|
| Frontend Framework | React 18 | UI library |
| Language | TypeScript | Type-safe development |
| Build Tool | Vite | Fast build and dev server |
| Styling | Tailwind CSS | Utility-first CSS |
| UI Components | Radix UI | Accessible component primitives |
| State Management | Zustand | Lightweight state management |
| Routing | React Router | Client-side routing |
| PDF Processing | PDF.js | PDF rendering |
| Canvas | HTML5 Canvas | Drawing and annotations |
| OCR | Tesseract.js | Client-side OCR |
| Backend Framework | Express.js | Node.js web framework |
| Real-time | Socket.IO | WebSocket communication |
| Database | PostgreSQL (Supabase) | Data persistence |
| Authentication | Supabase Auth | User authentication |
| File Storage | Supabase Storage | PDF file storage |
| AI Integration | Qwen3-VL, Ollama | AI vision processing |
| Browser Automation | Playwright | Automated takeoff |
| Email | Nodemailer | Email notifications |

### Deployment Process

**GitHub**: Code is pushed to GitHub repository (ubuildacademy/mcw-takeoff-tool)

**Vercel Auto-Deploy**: Vercel automatically builds and deploys frontend on push to main branch

**Railway Auto-Deploy**: Railway automatically builds and deploys backend on push to main branch

**Environment Variables**: Each platform has its own environment variables configured

**Database Migrations**: Supabase migrations are run manually or via CI/CD

## Environment Variables

**Frontend (Vercel)**:

• VITE_SUPABASE_URL

• VITE_SUPABASE_ANON_KEY

• VITE_API_BASE_URL (optional)

**Backend (Railway)**:

• PORT

• SUPABASE_URL

• SUPABASE_SERVICE_ROLE_KEY

• NODE_ENV

• ALLOWED_ORIGINS

• FRONTEND_URL

## Summary

Meridian Takeoff is a modern, scalable construction takeoff application built with a three-tier architecture. The frontend React application is hosted on Vercel, the backend Express.js API runs on Railway, and data is persisted in Supabase's PostgreSQL database. The system supports real-time collaboration, AI-powered features, and comprehensive reporting capabilities. The architecture is designed for scalability, with clear separation of concerns, RESTful API design, and modern web technologies. Continuous deployment is enabled through GitHub integration with both Vercel and Railway. **Key Architecture Benefits:** Scalable: Each tier can scale independently Reliable: Platform-managed hosting with automatic failover Secure: Row Level Security, JWT authentication, CORS protection Maintainable: Clear separation of concerns, TypeScript throughout Fast: CDN for static assets, optimized API responses

For questions or contributions, please refer to the GitHub repository:

https://github.com/ubuildacademy/mcw-takeoff-tool