

It all started with a memory of a beloved, discontinued piece of software.

“> “Ya see there’s this software I used to use called FabtrolMRP. I knew it inside and out. Well they were bought out by a larger company then simply dropped the product. I happen to have complete backups from 2012! Lets use Claude and MCP tools to dig in and reverse engineer this sucker. I miss it so much lol.”





Unearthing a Digital Fossil: The First Look Inside FabTrol MRP

The first prompt to an AI assistant: "look at my local folder... this is an old backup of the entire servers working directory. lets see how it works"

The screenshot shows a window titled "AI Assistant - Directory Analysis" with a list of files and directories. The list includes:

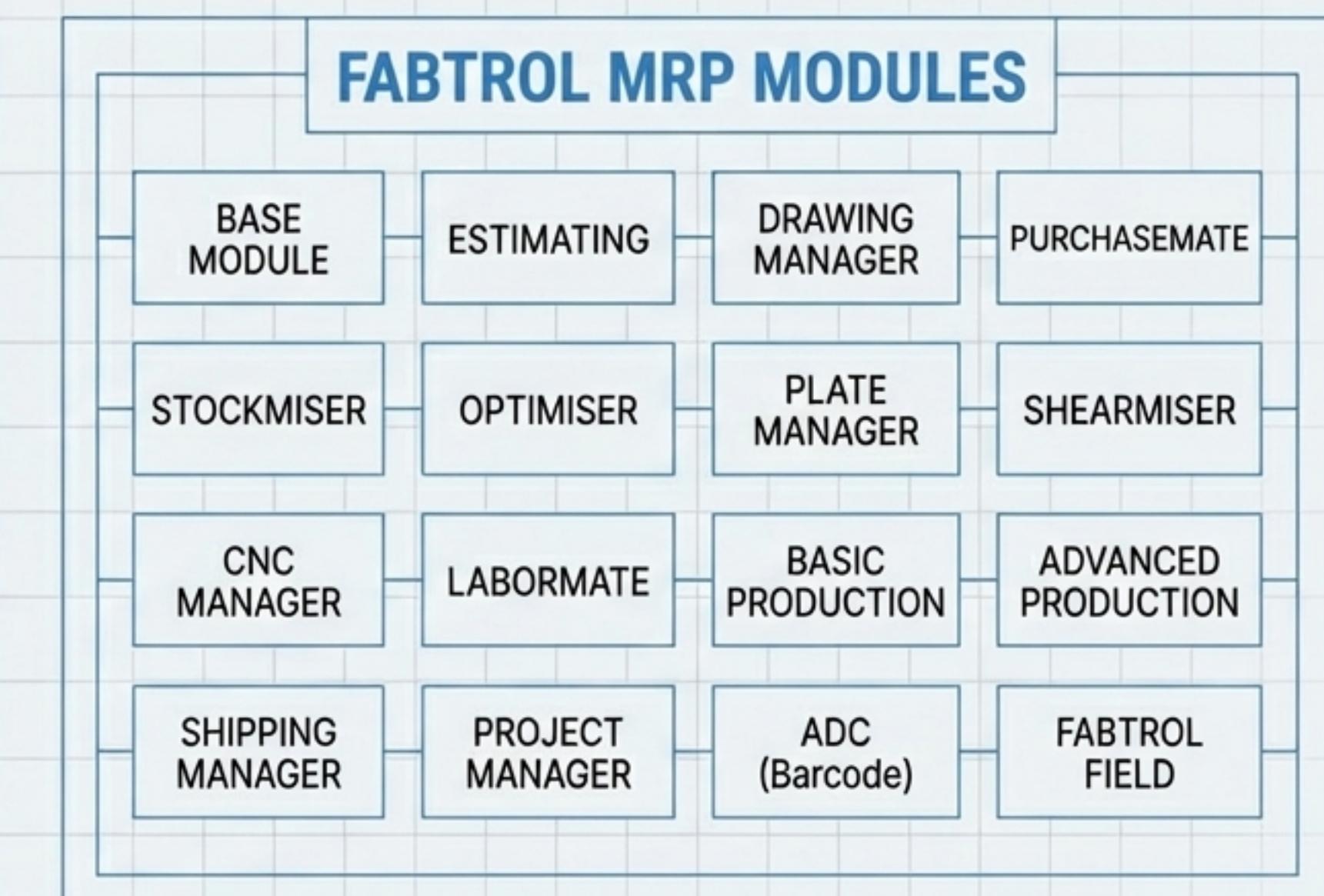
- [DIR] System - Core system files
- [DIR] Reports - Report templates and definitions
- [DIR] CNC - CNC integration files
- [DIR] DRAWINGS - Drawing storage
- [DIR] Parts - Parts database/files
- [DIR] Nesting output - Nesting results
- [DIR] Labels - Label templates (BarTender .BTW files)
- [DIR] Backups - System backups
- [DIR] Blds - Bldding/estimating data
-
- [FILE] account.dbf
- [FILE] bow.dbf
- [FILE] customer.dbf
- [FILE] job.dbf
- [FILE] part.dbf
- [FILE] supplier.dbf
- [FILE] vendor.dbf
- [FILE] fabtroldbc
- [FILE] fabtrol.dct
- [FILE] fabtrol.dcx
- [FILE] ftmrp.exe - Main Fabtrol executable
- [FILE] ftserverviews.sql
- [FILE] ftstoredprocedures.sql
- [FILE] custom1.prg
- [FILE] custom2.prg
- [FILE] custom3.prg
- [FILE] custom4.prg
- [FILE] custom5.prg
- [FILE] custom6.prg
- [FILE] custom7.prg
- [FILE] custom8.prg
- [FILE] custom9.prg
- [FILE] custom10.prg
- [FILE] fabtrol.ini
- [FILE] config.fpw

A callout box points to the bottom of the list with the text: "This is a treasure trove for understanding the original FabTrol architecture!"



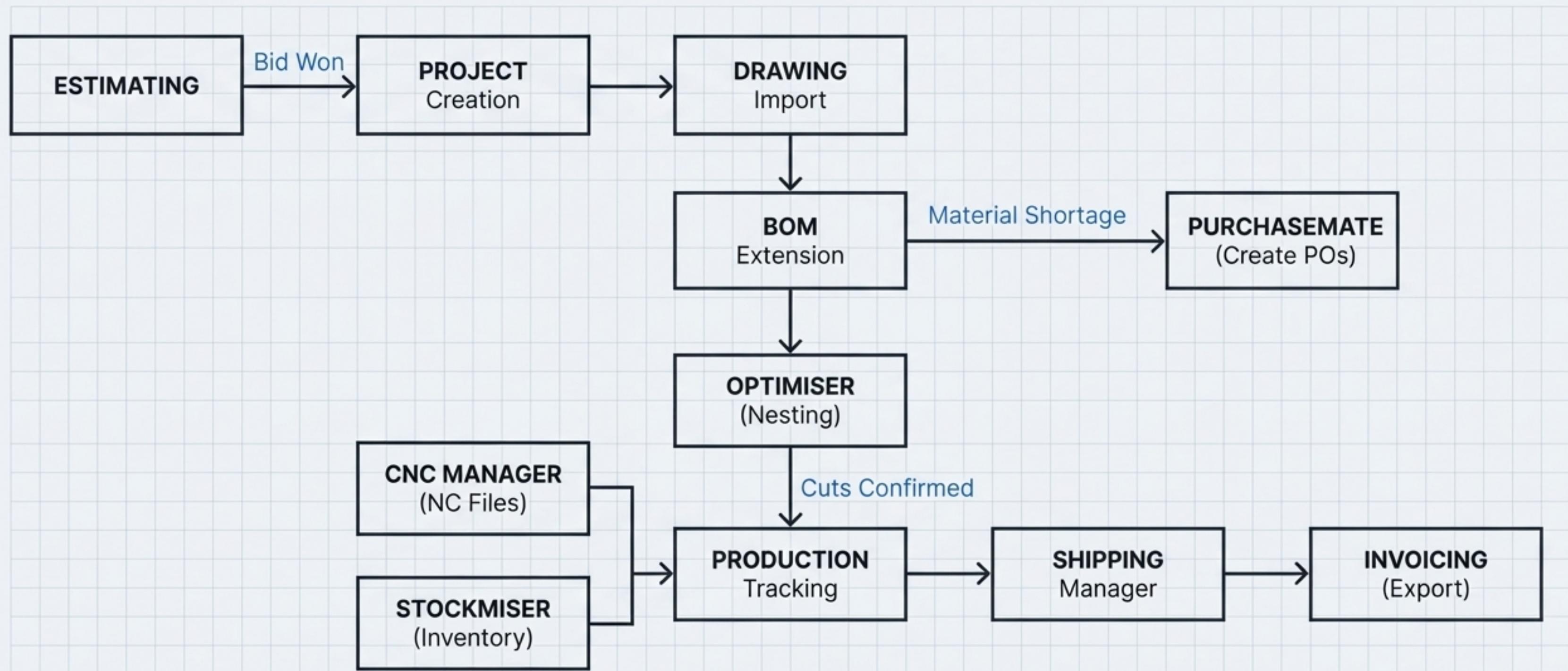
More Than Just Files: A Complete Manufacturing Ecosystem for Steel Fabrication.

Component	Technology
Core Platform	Microsoft Visual FoxPro 9.0
Database	DBF files / Microsoft SQL Server 2005/2008+
Report Engine	FoxPro FRX/FRT reports
Label Printing	BarTender (.BTW templates)
Metadata Framework	Stonefield Database Toolkit (SDT)



Built on Visual FoxPro 9.0, the system managed the entire fabrication process from estimating through shipping.

The Lifeblood of the System: Mapping the 'Quote to Ship' Workflow.



Decoding the Blueprint: The Core Data Model and Project Hierarchy.

PROJECT HIERARCHY	
PROJECT (Job)	
└ PPH_Project_Phase	
└ LOT_Project_Lot	
└ ASM_Assembly (Piece Marks – e.g., B1, C5)	
└ ASI_Assembly_Instance	
└ PRI_Part_Instance	
└ PWE_Part_WorkArea_Estimate	
└ BID_Base (Parts/components of assembly)	
└ DRW_Drawing	

Key Status Codes	
Status Field	Decoded Values
ASI_ProdPlanStatus	0=Not Started 10=In Progress 20=Complete
PWE_ProdStatus	0=Pending 10=In Progress 20=Complete
PBT_Status	0=Created 10=Released 20=Complete

From Reverse-Engineering to a Go-Forward Plan.

The analysis culminated in a comprehensive implementation guide for a modern replacement.

Recommended Technology Stack

 **Backend:** Laravel 11 (PHP)

 **Database:** MySQL 8 or PostgreSQL 15

 **Frontend:** Vue.js 3 + Tailwind CSS

 **Auth:** Office 365 OAuth



Migration Priority

1. Phase 1: Core Foundation (Users, Customers, Projects)
2. Phase 2: BOM Management (Import, Extension)
3. Phase 3: Purchasing & Inventory
4. Phase 4: Production
5. Phase 5: Nesting & CNC
6. Phase 6: Shipping

Forging the Future: Architecting 'SteelFlow MRP'.

```
steelflow/
  app/
    Models/
      Project.php
      Assembly.php
    Services/
      KissImporter.php
      LinearNestingEngine.php
  Http/
  database/
  resources/
    js/
    Pages/
```

CREATE TABLE projects

```
CREATE TABLE projects (
  id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  job_number VARCHAR(255) NOT NULL,
  name VARCHAR(255) NOT NULL,
  customer_id BIGINT UNSIGNED NOT NULL,
  status ENUM('pending', 'active', 'completed') NOT NULL DEFAULT 'pending',
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);
```

CREATE TABLE assemblies

```
CREATE TABLE assemblies (
  id BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
  project_id BIGINT UNSIGNED NOT NULL,
  mark VARCHAR(50) NOT NULL,
  description TEXT,
  quantity INT NOT NULL,
  weight_each_lbs DECIMAL(10, 2),
  status VARCHAR(50) NOT NULL,
  FOREIGN KEY (project_id) REFERENCES projects(id)
);
```

A modern Laravel/Vue.js monolith built for performance, scalability, and a user-friendly experience.

Preserving Critical Business Logic in Modern Code

WeightCalculator class

```
1 <?php
2
3 namespace App\Services;
4
5 class WeightCalculator
6 {
7     public function calculateImperial(float $quantity,
8         float $unitWeightLbs): float
9     {
10        return $quantity * $unitWeightLbs;
11    }
12
13    public function calculateMetric(float $quantity,
14        float $unitWeightLbs): float
15    {
16        // Convert lbs to kg: 1 lb = 0.453592 kg
17        $totalLbs = $this->calculateImperial($quantity,
18            $unitWeightLbs);
19        return $totalLbs * 0.453592;
20    }
21}
```

AssemblyStatus enum

```
1 <?php
2
3 namespace App\Enums;
4
5 enum AssemblyStatus: string
6 {
7     case NOT_STARTED = 'not_started';
8     case IN_PROGRESS = 'in_progress';
9     case ON_HOLD = 'on_hold';
10    case COMPLETED = 'completed';
11    case QA_CHECKED = 'qa_checked';
12    case SHIPPED = 'shipped';
13    case DELIVERED = 'delivered';
14}
```

NestingService class

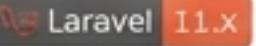
```
1 <?php
2
3 namespace App\Services;
4 use App\Enums\StockStatus;
5
6 class NestingService
7 {
8     public function approve(Stock $stock)
9     {
10        $stock->updateStatus(StockStatus::ASSIGNED);
11    }
12
13    public function verify(Stock $stock)
14    {
15        $stock->updateStatus(StockStatus::COMMITTED);
16    }
17
18    public function confirm(Stock $stock)
19    {
20        $stock->updateStatus(StockStatus::USED);
21    }
22}
23}
```

— Fidelity to original FabTrol logic ensures data integrity and operational consistency.

From Local Directory to Public Repository.

 meistro57 / **SteelFlow-MRP**

SteelFlow MRP is a modern, web-based manufacturing resource planning system designed specifically for the steel fabrication industry...

 License MIT 

Technology Stack

 Laravel  Vue.js  MySQL  Docker

 README

 Roadmap

- 1. Phase 1: Foundation
- 2. Phase 2: BOM Management
- 3. Phase 3: Purchasing
- ...

 commits commit history  1f3  ... 3 days ago

All Core Modules Implemented and Pushed.

A summary of progress from project initialization.

- ✓ **Phase 1: Foundation** (Docker, Auth, UI Shell)
- ✓ **Phase 2: BOM & Engineering** (Models, Weight Calc, KISS Importer)
- ✓ **Phase 3: Procurement** (Purchase Orders, Stock Items, Inventory Service)
- ✓ **Phase 4: Optimization (Nesting)** (Linear Nesting, Stock Logic)
- ✓ **Phase 5: Shop Floor Execution** (Work Area Routing, Time Tracking)
- ✓ **Phase 6: Shipping & Logistics** (Load Building, BOL Tracking)
- ✓ **Phase 7: Reporting & Polish** (Reporting Service, Dashboards)

CI/CD pipeline established with GitHub Actions for automated testing and builds.

The Vision Realized: The SteelFlow MRP Interface.

The screenshot displays the SteelFlow MRP interface with several key features highlighted:

- Real-time search:** A search bar at the top right of the dashboard.
- Dark/Light mode:** A toggle switch in the top right corner.
- Imperial/Metric toggle:** A switch located on the far right of the interface.

The main dashboard includes the following sections:

- Inventory List:** Shows a table of items with columns for Name, Description, Quantity, Status, and Location.
- Dashboard Metrics:** Displays Active Projects (12), Pending Orders (1), Picked Orders (39), and Resource Utilization (73.7%).
- KPI Charts:** Includes a line chart showing resource utilization over time and a bar chart for resource utilization by category.
- Projects:** A table listing multiple projects with columns for Name, Order ID, Status, Location, and Description.

I'm totally blown away right now lol

**“I mean I just started
this after lunch!”**



Session with Crush using Gemini 3 Flash Preview. Total spent so far? \$2.88