Phase 0 Module: Operations

Overview

The Operations module manages the catalog of operation types available in the shop. Operation types define what work can be done and at what cost - they're referenced when creating routings (Phase 1) and for job costing.

Priority: Must Have (Build after Parts, before Quotes)

Dependencies: None (foundational module)

Database Tables: `resource_groups` , `operation_types`

Terminology

Aligned with Contour's legacy system for familiarity:

Jigged Term	Legacy System Term	Description
Resource Group	Resource Group	A category of related operations (e.g., "LATHE&MILL", "CNC", "EDM", "Hone")
Operation Type	Operation Type	A specific operation or machine with labor rate (e.g., "HURCO Mill" @ $135/hr)

Future Scalability: This two-level model can extend to support:

- Manufacturing Lines (ordered sequences of operations)
- Buildings / Sites (physical locations)
- Machine Instances (multiple identical machines)

User Stories

As a...	I want to...	So that...
Owner/Admin	View all operation types organized by resource group	I can see our shop's capabilities at a glance

As a...	I want to...	So that...
Owner/Admin	Create resource groups to categorize operations	I can organize similar machines/operations together
Owner/Admin	Create operation types with hourly labor rates	I can track costs accurately
Owner/Admin	Edit operation type rates when costs change	My quotes and job costing stay accurate
Owner/Admin	Bulk import operation types from my legacy system	I can migrate quickly without manual data entry
Owner/Admin	Add custom fields to operation types	I can track setup time, capabilities, or other shop-specific data
Estimator	Look up operation type rates when quoting	I can calculate accurate job costs
Estimator	See operation types sorted by resource group	I can quickly find the right operation

Data Model

Resource Groups Table (`resource_groups`)

Field	Type	Required	Description
id	UUID	Yes	Primary key (auto-generated)
company_id	UUID (FK)	Yes	Link to company (multi-tenant isolation)
name	Text	Yes	Group name (e.g., "LATHE&MILL", "CNC", "Hone")
description	Text	No	Optional description
display_order	Integer	No	Sort order in UI (default: 0)
created_at	Timestamp	Yes	Auto-generated
updated_at	Timestamp	Yes	Auto-updated on changes

Unique Constraint: `(company_id, name)`

Operation Types Table (`operation_types`)

Field	Type	Required	Description
id	UUID	Yes	Primary key (auto-generated)
company_id	UUID (FK)	Yes	Link to company (multi-tenant isolation)
resource_group_id	UUID (FK)	No	Link to resource group (NULL = ungrouped)
name	Text	Yes	Operation type name (e.g., "HURCO Mill", "Mazak Lathe")
code	Text	No	Short code for display (e.g., "HRC-M1")
labor_rate	Decimal	No	Hourly rate in dollars (e.g., 135.00)
description	Text	No	Optional description or notes
metadata	JSONB	No	Flexible field for future extensions (see below)
created_at	Timestamp	Yes	Auto-generated
updated_at	Timestamp	Yes	Auto-updated on changes

Unique Constraint: `(company_id, name)`

Metadata JSONB Structure

The `metadata` column provides flexibility for shop-specific needs without schema changes:

```
{
  "setup_time_minutes": 30,
  "capabilities": ["5-axis", "high-speed"],
  "max_part_size": { "x": 24, "y": 18, "z": 12 },
  "manufacturer": "Hurco",
  "model": "VM10i",
  "year_acquired": 2019,
  "notes": "Preferred for aluminum parts"
}
```

UI Screens

1. Operations List (Grouped View)

Route: `/dashboard/{companyId}/operations`

Layout: Accordion-style grouped list (not a flat table)

Features:

- Resource groups as expandable sections
- Operation types listed within each group
- "Ungrouped" section at bottom for orphan operation types
- Search box (searches operation type name and code)
- "+ New Operation Type" button
- "+ New Group" button (secondary)
- Click operation type row to view/edit

Group Header Display:

```
▼ LATHE&MILL (7 operation types)            [Edit Group]
  ├─ LATHE           $105/hr    [Edit]
  ├─ SAW             $90/hr     [Edit]
  ├─ BRIDGEPORT MILL   $95/hr     [Edit]
  ├─ HURCO Lathe      $135/hr    [Edit]
  └─ Mazak Lathe      $135/hr    [Edit]

▼ CNC (2 operation types)                   [Edit Group]
  ├─ POLISH          $95/hr     [Edit]
  └─ HURCO Mill       $135/hr    [Edit]

▼ Ungrouped (5 operation types)
  ├─ Hone            $95/hr     [Edit]
  └─ TRM30 Mill       $135/hr    [Edit]
```

Empty State:

"No operation types yet. Create your first operation type or import from your legacy system."

2. Operation Type Create/Edit

Route: `/dashboard/{companyId}/operations/new` or `/dashboard/{companyId}/operations/{id}/edit`

Form Sections:

▸ **Basic Information**

- Name (required)
- Code (optional, for short display)
- Resource Group (dropdown, optional)
- Description

▸ **Costing**

- Labor Rate ($/hour)

▸ **Additional Details** (collapsible, optional)

- Setup Time (minutes) - stored in metadata
- Notes - stored in metadata

Actions:

- Save → Returns to list
- Cancel → Returns to list without saving
- Delete (edit mode only) → Confirmation dialog

Delete Validation:

- Cannot delete if operation type is used in any routing (Phase 1)
- Show warning with count of affected routings

3. Resource Group Create/Edit

Route: Modal overlay (not separate page)

Trigger: "+ New Group" button or "Edit Group" link

Modal Content:

- Name (required)
- Description (optional)
- Display Order (number, for sorting groups)

Actions:

- Save → Closes modal, refreshes list
- Cancel → Closes modal
- Delete (edit mode) → Moves all operation types to "Ungrouped", deletes group

---

AI-Powered Bulk Import

Route: `/dashboard/{companyId}/operations/import`

Uses the same AI-powered import infrastructure as Customers and Parts.

Import Flow

1. **Upload CSV/Excel** - Parse file, extract headers + sample rows
2. **AI Analysis** - AI suggests column mappings with confidence scores
3. **Review Mappings** - User confirms/adjusts mappings
4. **Group Handling:**
    - Auto-create resource groups from unique values in mapped group column
    - Option to map to existing groups
    - Option to import all as ungrouped
5. **Execute** - Import with results summary

Expected Source Columns (from Contour export)

Source Column	Maps To	Notes
name	operation_types.name	Primary identifier
laborRate	operation_types.labor_rate	Hourly rate

Source Column	Maps To	Notes
resourceGroup	resource_groups.name	Auto-creates groups
resource	operation_types.code	Optional short code
_id	operation_types.metadata.legacy_id	Preserve for reference

Conflict Detection

- **Duplicate name** within company → Conflict (skip or update)
- **Missing laborRate** → Warning (import with NULL)
- **Unknown resourceGroup** → Auto-create new group

API Architecture

CRUD Operations → Direct Supabase Calls

Simple create, read, update, delete operations use direct Supabase client calls from the frontend (via `utils/operationsAccess.ts`). This follows the same pattern as Customers and Parts:

- Avoids Vercel serverless cold starts
- Leverages Supabase RLS for security
- Simpler, faster for basic operations

Complex Operations → FastAPI Endpoints

AI-powered features that require server-side processing use FastAPI:

- `POST /api/operations/import/analyze` - AI mapping suggestions
- `POST /api/operations/import/execute` - Perform import with group auto-creation

Rate Lookup Logic

When calculating job costs (Phase 1+), the system will:

```
function getOperationTypeRate(operationTypeId: string): number | null {
  const operationType = await getOperationType(operationTypeId);
  return operationType?.labor_rate ?? null;
}
```

Acceptance Criteria

Resource Groups

- [ ] Can view list of resource groups
- [ ] Can create new resource group
- [ ] Can edit resource group name/description
- [ ] Can reorder groups via display_order
- [ ] Can delete empty group
- [ ] Deleting group with operation types moves them to "Ungrouped"
- [ ] Group names are unique within company

Operation Types

- [ ] Can view operation types organized by group
- [ ] Can search operation types by name or code
- [ ] Can create new operation type with group assignment
- [ ] Can create operation type without group (ungrouped)
- [ ] Can edit operation type details
- [ ] Can change operation type's group
- [ ] Can set labor rate
- [ ] Operation type names are unique within company
- [ ] Form shows validation errors inline

AI-Powered Import

- [ ] Can upload CSV or Excel file
- [ ] AI analyzes file and suggests column mappings

- [ ] Confidence scores displayed with color coding
- [ ] Resource groups auto-created from import data
- [ ] Detects duplicate operation type names
- [ ] Can skip conflicts and import valid rows
- [ ] Shows import results: imported, skipped, groups created
- [ ] Legacy IDs preserved in metadata

Database Migration

Note: If tables were previously created as `resources`, run this migration to rename:

```
-- Rename resources table to operation_types
ALTER TABLE resources RENAME TO operation_types;

-- Update foreign key column name for clarity (optional)
ALTER TABLE operation_types RENAME COLUMN resource_group_id TO reso
urce_group_id;

-- Update indexes
ALTER INDEX idx_resources_company RENAME TO idx_operation_types_com
pany;
ALTER INDEX idx_resources_group RENAME TO idx_operation_types_group;

-- Update RLS policy
ALTER POLICY resources_company_isolation ON operation_types RENAME TO
operation_types_company_isolation;
```

Fresh install schema:

```
-- Resource Groups (categories)
CREATE TABLE resource_groups (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  company_id UUID NOT NULL REFERENCES companies(id) ON DELETE CASC
ADE,
  name TEXT NOT NULL,
  description TEXT,
  display_order INTEGER DEFAULT 0,
  created_at TIMESTAMPTZ DEFAULT now() NOT NULL,
  updated_at TIMESTAMPTZ DEFAULT now() NOT NULL,
  UNIQUE(company_id, name)
);

-- Operation Types
CREATE TABLE operation_types (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  company_id UUID NOT NULL REFERENCES companies(id) ON DELETE CASC
ADE,
  resource_group_id UUID REFERENCES resource_groups(id) ON DELETE SET
NULL,
  name TEXT NOT NULL,
  code TEXT,
  labor_rate DECIMAL(10,2),
  description TEXT,
  metadata JSONB DEFAULT '{}',
  created_at TIMESTAMPTZ DEFAULT now() NOT NULL,
  updated_at TIMESTAMPTZ DEFAULT now() NOT NULL,
  UNIQUE(company_id, name)
);

-- RLS Policies
ALTER TABLE resource_groups ENABLE ROW LEVEL SECURITY;
ALTER TABLE operation_types ENABLE ROW LEVEL SECURITY;

CREATE POLICY resource_groups_company_isolation ON resource_groups
  USING (company_id IN (
    SELECT company_id FROM user_company_access WHERE user_id = auth.ui
d()
  ));

CREATE POLICY operation_types_company_isolation ON operation_types
```

```
  USING (company_id IN (
    SELECT company_id FROM user_company_access WHERE user_id = auth.uid()
  ));

-- Indexes
CREATE INDEX idx_resource_groups_company ON resource_groups(company_id);
CREATE INDEX idx_operation_types_company ON operation_types(company_id);
CREATE INDEX idx_operation_types_group ON operation_types(resource_group_id);
```