Phase 3: Sales Capture Module - Complete Documentation

Overview

The Sales Capture Module is a comprehensive Point of Sale (POS) and inventory management system integrated into CitaPlanner. It enables businesses to manage products, track inventory, process sales, calculate commissions, and generate detailed reports.

Table of Contents

1. Database Schema
2. API Endpoints
3. Business Logic Services
4. User Interface Components
5. Features
6. Deployment Instructions
7. Usage Guide

Database Schema

New Tables

1. product_categories

Organizes products into categories for better management.

```
model ProductCategory {
  id          String    @id @default(cuid())
  name        String
  description String?
  isActive    Boolean   @default(true)
  tenantId    String
  createdAt   DateTime  @default(now())
  updatedAt   DateTime  @updatedAt

  products    Product[]
}
```

2. suppliers

Manages supplier information for product sourcing.

```
model Supplier {
  id            String    @id @default(cuid())
  name          String
  contactPerson String?
  phone         String?
  email         String?
  address       String?
  notes         String?
  isActive      Boolean   @default(true)
  tenantId      String
  createdAt     DateTime  @default(now())
  updatedAt     DateTime  @updatedAt

  products      Product[]
}
```

3. products

Stores product information including inventory levels and pricing.

```
model Product {
  id          String      @id @default(cuid())
  name        String
  description String?
  sku         String?
  type        ProductType @default(SALE)      // SALE or INTERNAL
  unit        ProductUnit @default(PIECE)     // PIECE or SERVICE
  stock       Float       @default(0)
  minStock    Float       @default(0)
  costPrice   Float       @default(0)
  salePrice   Float       @default(0)
  isActive    Boolean     @default(true)
  tenantId    String
  supplierId  String?
  categoryId  String?
  createdAt   DateTime    @default(now())
  updatedAt   DateTime    @updatedAt
}
```

4. sales

Records all sales transactions.

```
model Sale {
  id            String            @id @default(cuid())
  saleNumber    String            @unique
  subtotal      Float
  discount      Float             @default(0)
  tax           Float             @default(0)
  total         Float
  paymentMethod SalePaymentMethod @default(CASH)
  status        SaleStatus        @default(COMPLETED)
  saleDate      DateTime          @default(now())
  notes         String?
  tenantId      String
  clientId      String?
  userId        String
  createdAt     DateTime          @default(now())
  updatedAt     DateTime          @updatedAt

  saleItems     SaleItem[]
}
```

5. sale_items

Individual items (products or services) within a sale.

```
model SaleItem {
  id               String  @id @default(cuid())
  itemType         String  // "SERVICE" or "PRODUCT"
  quantity         Float
  unitPrice        Float
  subtotal         Float
  commissionRate   Float?
  commissionAmount Float?
  saleId           String
  serviceId        String?
  productId        String?
  professionalId   String?
  createdAt        DateTime @default(now())
  updatedAt        DateTime @updatedAt
}
```

6. inventory_movements

Tracks all inventory changes (in, out, adjustments).

```
model InventoryMovement {
  id            String        @id @default(cuid())
  movementType  MovementType  // IN, OUT, ADJUSTMENT
  quantity      Float
  reason        String?
  reference     String?
  tenantId      String
  productId     String
  userId        String
  createdAt     DateTime      @default(now())
  updatedAt     DateTime      @updatedAt
}
```

7. professional_commissions

Aggregates commissions by professional and period.

```
model ProfessionalCommission {
  id                String          @id @default(cuid())
  period            String          // Format: "YYYY-MM"
  totalSales        Float           @default(0)
  totalCommissions  Float           @default(0)
  status            CommissionStatus @default(PENDING)
  paidDate          DateTime?
  notes             String?
  tenantId          String
  professionalId    String
  createdAt         DateTime        @default(now())
  updatedAt         DateTime        @updatedAt
}
```

New Enums

```
enum ProductType {
  SALE
  INTERNAL
}

enum ProductUnit {
  PIECE
  SERVICE
}

enum SaleStatus {
  PENDING
  COMPLETED
  CANCELLED
}

enum SalePaymentMethod {
  CASH
  CARD
  TRANSFER
  OTHER
}

enum MovementType {
  IN
  OUT
  ADJUSTMENT
}

enum CommissionStatus {
  PENDING
  PAID
}
```

API Endpoints

Products

`GET /api/products`

Get all products with optional filters.

Query Parameters:
- `type`: ProductType (SALE, INTERNAL)
- `categoryId`: Filter by category
- `supplierId`: Filter by supplier
- `isActive`: Filter by active status
- `lowStock`: Show only low stock products

Response:

```json
[
  {
    "id": "prod_123",
    "name": "Shampoo Premium",
    "sku": "SHP-001",
    "type": "SALE",
    "stock": 15,
    "minStock": 10,
    "salePrice": 250.00,
    "category": { "name": "Hair Care" },
    "supplier": { "name": "Beauty Supplies Inc" }
  }
]
```

`POST /api/products`

Create a new product.

Request Body:

```json
{
  "name": "Shampoo Premium",
  "description": "Professional hair shampoo",
  "sku": "SHP-001",
  "type": "SALE",
  "unit": "PIECE",
  "stock": 20,
  "minStock": 10,
  "costPrice": 150.00,
  "salePrice": 250.00,
  "categoryId": "cat_123",
  "supplierId": "sup_456"
}
```

`GET /api/products/[id]`

Get product details including recent inventory movements.

`PUT /api/products/[id]`

Update product information.

`DELETE /api/products/[id]`

Soft delete a product (sets isActive to false).

`GET /api/products/low-stock`

Get all products with stock at or below minimum stock level.

Suppliers

`GET /api/suppliers`

Get all suppliers.

`POST /api/suppliers`

Create a new supplier.

`GET /api/suppliers/[id]`

Get supplier details with associated products.

`PUT /api/suppliers/[id]`

Update supplier information.

`DELETE /api/suppliers/[id]`

Soft delete a supplier.

Product Categories

`GET /api/product-categories`

Get all product categories.

`POST /api/product-categories`

Create a new category.

Sales

`GET /api/sales`

Get all sales with optional filters.

Query Parameters:
- `startDate`: Filter by start date
- `endDate`: Filter by end date
- `clientId`: Filter by client
- `userId`: Filter by seller
- `status`: Filter by status
- `paymentMethod`: Filter by payment method

Response:

```json
[
  {
    "id": "sale_123",
    "saleNumber": "20251008001",
    "total": 850.00,
    "paymentMethod": "CASH",
    "status": "COMPLETED",
    "saleDate": "2025-10-08T10:30:00Z",
    "client": {
      "firstName": "Juan",
      "lastName": "Pérez"
    },
    "saleItems": [
      {
        "itemType": "SERVICE",
        "service": { "name": "Haircut" },
        "quantity": 1,
        "unitPrice": 350.00,
        "professional": {
          "firstName": "María",
          "lastName": "García"
        },
        "commissionAmount": 52.50
      },
      {
        "itemType": "PRODUCT",
        "product": { "name": "Shampoo Premium" },
        "quantity": 2,
        "unitPrice": 250.00
      }
    ]
  }
]
```


`POST /api/sales`

Create a new sale.

Request Body:

```json
{
  "clientId": "client_123",
  "items": [
    {
      "itemType": "SERVICE",
      "serviceId": "service_456",
      "professionalId": "user_789",
      "quantity": 1,
      "unitPrice": 350.00,
      "commissionRate": 15
    },
    {
      "itemType": "PRODUCT",
      "productId": "prod_123",
      "quantity": 2,
      "unitPrice": 250.00
    }
  ],
  "discount": 0,
  "tax": 0,
  "paymentMethod": "CASH",
  "notes": "Cliente frecuente"
}
```

Features:
- Automatically generates unique sale number
- Validates product stock availability
- Deducts product inventory automatically
- Calculates and records commissions for services
- Creates inventory movement records

`GET /api/sales/[id]`

Get sale details with all items.

`POST /api/sales/[id]/cancel`

Cancel a sale.

Features:
- Restores product inventory
- Reverses commission calculations
- Updates sale status to CANCELLED

Inventory

`GET /api/inventory/movements`

Get inventory movements with filters.

Query Parameters:
- `productId` : Filter by product
- `movementType` : Filter by type (IN, OUT, ADJUSTMENT)
- `startDate` : Filter by start date
- `endDate` : Filter by end date

`POST /api/inventory/adjustment`

Create a stock adjustment.

Request Body:

```json
{
  "productId": "prod_123",
  "newStock": 25,
  "reason": "Physical inventory count"
}
```

`GET /api/inventory/stock-report`

Get comprehensive stock report with values and status.

Response:

```json
[
  {
    "id": "prod_123",
    "name": "Shampoo Premium",
    "stock": 8,
    "minStock": 10,
    "stockStatus": "LOW",
    "costPrice": 150.00,
    "stockValue": 1200.00,
    "category": { "name": "Hair Care" }
  }
]
```

Commissions

`GET /api/commissions`

Get commissions with filters.

Query Parameters:
- `professionalId` : Filter by professional
- `period` : Filter by period (YYYY-MM)
- `status` : Filter by status (PENDING, PAID)

`POST /api/commissions/[id]/pay`

Mark commission as paid.

Request Body:

```json
{
  "notes": "Paid via bank transfer"
}
```

`GET /api/commissions/summary`

Get commission summary for a professional.

Query Parameters:
- `professionalId` : Required
- `period` : Optional (YYYY-MM)

Response:


```json
{
  "totalPending": 1250.00,
  "totalPaid": 3500.00,
  "totalSales": 15000.00,
  "byPeriod": [
    {
      "period": "2025-10",
      "totalSales": 5000.00,
      "totalCommissions": 750.00,
      "status": "PENDING"
    }
  ]
}
```

Reports

`GET /api/reports/sales`

Get comprehensive sales report.

Query Parameters:
- `startDate` : Required
- `endDate` : Required
- `groupBy` : Optional (day, week, month)
- `productId` : Optional
- `serviceId` : Optional
- `professionalId` : Optional
- `clientId` : Optional
- `paymentMethod` : Optional

Response:


```json
{
  "summary": {
    "totalSales": 45000.00,
    "totalDiscount": 500.00,
    "totalTax": 0,
    "salesCount": 125,
    "averageSale": 360.00
  },
  "groupedData": [
    {
      "period": "2025-10-01",
      "total": 1500.00,
      "count": 5
    }
  ],
  "sales": [...]
}
```

`GET /api/reports/top-products`

Get top-selling products.

Query Parameters:
- `startDate` : Required
- `endDate` : Required
- `limit` : Optional (default: 10)

`GET /api/reports/top-services`

Get top-performing services.

`GET /api/reports/professional-performance`

Get performance metrics by professional.

Dashboard

`GET /api/dashboard/sales`

Get sales dashboard metrics.

Query Parameters:
- `period` : day, week, month (default: month)

Response:

```json
{
  "period": "month",
  "salesSummary": {
    "totalSales": 45000.00,
    "salesCount": 125,
    "averageSale": 360.00
  },
  "salesTrend": [...],
  "topProducts": [...],
  "topServices": [...],
  "lowStockAlerts": 5,
  "lowStockProducts": [...],
  "pendingCommissions": {
    "count": 8,
    "total": 3500.00,
    "items": [...]
  }
}
```

Business Logic Services

ProductService

Located at: `src/services/productService.ts`

Key Methods:
- `createProduct(data)` : Create new product
- `getProducts(tenantId, filters)` : Get products with filters

- `getProductById(id, tenantId)`: Get product details
- `updateProduct(id, tenantId, data)`: Update product
- `deleteProduct(id, tenantId)`: Soft delete product
- `getLowStockProducts(tenantId)`: Get low stock alerts
- `validateStock(productId, quantity)`: Check stock availability
- `updateStock(productId, quantity, operation)`: Update stock levels

SaleService

Located at: `src/services/saleService.ts`

Key Methods:
- `createSale(data)`: Create new sale with automatic inventory and commission processing
- `getSales(tenantId, filters)`: Get sales with filters
- `getSaleById(id, tenantId)`: Get sale details
- `cancelSale(id, tenantId, userId)`: Cancel sale and reverse transactions
- `generateSaleNumber(tenantId)`: Generate unique sale number

Sale Creation Process:
1. Validates product stock availability
2. Generates unique sale number (format: YYYYMMDD####)
3. Calculates totals and commissions
4. Creates sale and items in transaction
5. Deducts product inventory
6. Records inventory movements
7. Records professional commissions

InventoryService

Located at: `src/services/inventoryService.ts`

Key Methods:
- `createMovement(data)`: Record inventory movement
- `getMovements(tenantId, filters)`: Get movement history
- `createAdjustment(tenantId, productId, newStock, reason, userId)`: Adjust stock levels
- `getStockReport(tenantId)`: Generate stock report with values

CommissionService

Located at: `src/services/commissionService.ts`

Key Methods:
- `recordCommission(data)`: Record commission for a sale
- `getCommissions(tenantId, filters)`: Get commissions with filters
- `markAsPaid(id, tenantId, notes)`: Mark commission as paid
- `getCommissionSummary(tenantId, professionalId, period)`: Get summary by professional

Commission Tracking:
- Automatically aggregates by professional and period (YYYY-MM)
- Supports both positive and negative amounts (for cancellations)
- Tracks pending vs paid status

ReportService

Located at: `src/services/reportService.ts`

Key Methods:
- `getSalesReport(tenantId, filters)`: Comprehensive sales report
- `getTopProducts(tenantId, startDate, endDate, limit)`: Top products by revenue
- `getTopServices(tenantId, startDate, endDate, limit)`: Top services by revenue
- `getProfessionalPerformance(tenantId, startDate, endDate)`: Performance by professional

User Interface Components

POS System

Location: `app/sales/pos/page.tsx`

Features:
- Quick product/service search and selection
- Client selection with integration to CRM
- Professional selection for services
- Real-time total calculation
- Multiple payment methods
- Commission rate configuration per service
- Sale completion and receipt generation

Workflow:
1. Select client (optional)
2. Add products/services to cart
3. For services: select professional and commission rate
4. Apply discounts/taxes if needed
5. Select payment method
6. Complete sale

Product Management

Location: `app/inventory/products/page.tsx`

Features:
- Product list with filters (type, category, supplier, stock status)
- Add/edit product form
- Stock level indicators with color coding
- Low stock alerts
- Quick actions (edit, delete, view details)
- Bulk operations

Inventory Control

Location: `app/inventory/movements/page.tsx`

Features:
- Movement history with filters
- Stock adjustment form
- Movement type indicators (IN, OUT, ADJUSTMENT)
- Reference tracking (sale numbers, etc.)
- Export to Excel/PDF

Supplier Management

Location: `app/inventory/suppliers/page.tsx`

Features:
- Supplier list with contact information
- Add/edit supplier form
- Associated products view
- Purchase history (future enhancement)

Sales Reports

Location: `app/reports/sales/page.tsx`

Features:
- Multiple report types:
- Sales by period
- Sales by product/service
- Sales by professional
- Sales by client
- Sales by payment method
- Date range selection
- Grouping options (day, week, month)
- Charts and visualizations
- Export to PDF/Excel

Sales Dashboard

Location: `app/dashboard/sales/page.tsx`

Features:
- Key metrics cards:
- Daily/weekly/monthly sales
- Average sale value
- Total transactions
- Sales trend chart
- Top products/services charts
- Revenue analysis
- Pending commissions summary
- Low inventory alerts
- Quick actions

Commission Management

Location: `app/commissions/page.tsx`

Features:
- Commission list by professional
- Period selection (monthly)
- Status filters (pending, paid)
- Mark as paid functionality
- Commission reports
- Export capabilities

Features

1. Product & Inventory Management

- Two product types: SALE (for selling) and INTERNAL (for internal use)
- SKU tracking
- Stock level monitoring
- Minimum stock alerts
- Cost and sale price tracking
- Category and supplier organization
- Inventory movement history

2. Point of Sale (POS)

- Quick sale registration
- Mixed sales (products + services)
- Client association
- Multiple payment methods
- Discount and tax support
- Automatic inventory deduction
- Commission calculation
- Unique sale number generation

3. Commission System

- Automatic commission calculation for services
- Configurable commission rates per service
- Monthly aggregation by professional
- Pending/paid status tracking
- Commission reports
- Payment recording

4. Inventory Tracking

- Automatic stock deduction on sales
- Manual stock adjustments
- Movement history (IN, OUT, ADJUSTMENT)
- Reason and reference tracking
- Stock value calculation

5. Reporting & Analytics

- Sales reports with multiple filters
- Top products/services analysis
- Professional performance metrics
- Revenue and profitability analysis
- Trend analysis with charts
- Export capabilities

6. Dashboard

- Real-time metrics

- Sales trends visualization
- Low stock alerts
- Pending commissions summary
- Quick access to key functions

Deployment Instructions

1. Database Migration

After merging this PR, run the Prisma migration in your production environment:

```
# SSH into your Easypanel server or access the app container

# Navigate to the app directory
cd /path/to/app

# Run Prisma migration
npx prisma migrate deploy

# Generate Prisma client
npx prisma generate

# Restart the application
pm2 restart citaplanner
# OR if using Docker
docker-compose restart
```

2. Environment Variables

Ensure these environment variables are set (should already be configured):

```
DATABASE_URL="postgresql://user:password@host:port/database"
NEXTAUTH_SECRET="your-secret-key"
NEXTAUTH_URL="https://your-domain.com"
```

3. Verification Steps

After deployment, verify the following:

1. **Database Tables Created:**
   `sql`
   ```
   SELECT table_name FROM information_schema.tables
   WHERE table_schema = 'public'
   AND table_name IN (
     'products', 'suppliers', 'product_categories',
     'sales', 'sale_items', 'inventory_movements',
     'professional_commissions'
   );
   ```

2. **API Endpoints Accessible:**
   - Test: `GET /api/products`
   - Test: `GET /api/sales`
   - Test: `GET /api/dashboard/sales`

3. **UI Pages Load:**
   - Visit: `/sales/pos`
   - Visit: `/inventory/products`
   - Visit: `/dashboard/sales`

4. Initial Data Setup

Optionally, create initial data:

1. **Product Categories:**
   - Navigate to Product Categories
   - Create categories for your business

2. **Suppliers:**
   - Navigate to Suppliers
   - Add your suppliers

3. **Products:**
   - Navigate to Products
   - Add your initial product catalog

Usage Guide

Creating Your First Sale

1. **Navigate to POS:**
   - Go to Sales > Point of Sale

2. **Select Client (Optional):**
   - Search and select existing client
   - Or proceed without client

3. **Add Items:**
   - Search for products/services
   - Click to add to cart
   - For services: select professional and commission rate
   - Adjust quantities as needed

4. **Apply Discounts/Taxes:**
   - Enter discount amount if applicable
   - Enter tax amount if applicable

5. **Select Payment Method:**
   - Choose: Cash, Card, Transfer, or Other

6. **Complete Sale:**
   - Review total
   - Click “Complete Sale”
   - Print/view receipt

Managing Inventory

1. **Add New Product:**
   - Go to Inventory > Products

  - Click “Add Product”
  - Fill in details (name, SKU, prices, stock)
  - Select category and supplier
  - Set minimum stock level
  - Save

2. **Adjust Stock:**
  - Go to Inventory > Movements
  - Click “Stock Adjustment”
  - Select product
  - Enter new stock level
  - Provide reason
  - Save

3. **Monitor Low Stock:**
  - Check dashboard for alerts
  - Or go to Products and filter by “Low Stock”
  - Reorder from suppliers as needed

Viewing Reports

1. **Sales Report:**
  - Go to Reports > Sales
  - Select date range
  - Choose grouping (day/week/month)
  - Apply filters (product, service, professional, client)
  - View charts and data
  - Export if needed

2. **Commission Report:**
  - Go to Commissions
  - Select professional
  - Choose period
  - View pending and paid commissions
  - Mark as paid when processed

3. **Dashboard Overview:**
  - Go to Dashboard > Sales
  - Select period (day/week/month)
  - View key metrics and trends
  - Check alerts and pending items

Integration with Existing Modules

Phase 1 Integration (Services)

- Sales can include services from the service catalog
- Service prices are used in sales
- Commission rates can be configured per service

Phase 2 Integration (CRM)

- Sales can be associated with clients from CRM
- Client purchase history is tracked
- Client preferences can influence sales

Multi-tenant Architecture

- All data is properly scoped by tenantId
- Branch-level reporting (future enhancement)
- User role-based access control

Security Considerations

1. **Authentication:**
   - All API endpoints require valid session
   - User must belong to the tenant

2. **Authorization:**
   - Role-based access control (future enhancement)
   - Sensitive operations (cancel sale, mark commission paid) should be restricted

3. **Data Validation:**
   - Stock validation before sale completion
   - Price and quantity validation
   - Tenant isolation enforced

4. **Audit Trail:**
   - All inventory movements are logged
   - Sale history is immutable (cancellations create new records)
   - Commission changes are tracked

Future Enhancements

1. **Purchase Orders:**
   - Create purchase orders to suppliers
   - Track order status
   - Automatic stock updates on receipt

2. **Barcode Scanning:**
   - Barcode generation for products
   - Scanner integration in POS

3. **Multi-currency Support:**
   - Support for multiple currencies
   - Exchange rate management

4. **Advanced Reporting:**
   - Profit margin analysis

   - Inventory turnover rates
   - Seasonal trends

5. **Mobile POS:**
   - Mobile-optimized POS interface
   - Offline mode with sync

6. **Loyalty Program:**
   - Points system
   - Rewards tracking
   - Integration with sales

Support

For issues or questions:
1. Check this documentation
2. Review API endpoint documentation
3. Check database schema
4. Contact development team

Changelog

Version 1.0.0 (Phase 3 Initial Release)

- Complete POS system
- Product and inventory management
- Supplier management
- Commission tracking
- Sales reporting
- Dashboard with key metrics
- Integration with Phase 1 and Phase 2 modules

Last Updated: October 8, 2025
Version: 1.0.0
Status: Ready for Production