Van Dyk Equipment Manager - Modular Architecture Guide

📁 File Structure

```
equipment_manager/
├── app.py                  # 🏗️ Master application (main entry point)
├── shared_config.py        # ⚙️ Shared configuration and utilities
├── equipment_manager.py    # 📝 Equipment add/edit functionality
├── search_equipment.py     # 🔍 Search and analysis functionality
├── network_visualization.py # 🌐 Network visualization module
├── db_utils.py             # 💾 Database utilities (existing)
├── validation.py           # ✅ Data validation (existing)
├── requirements.txt        # 📦 Package dependencies
└── logs/                   # 📊 Application logs
    └── app.log
```

🚀 How to Run

```bash
# Navigate to project directory
cd equipment_manager

# Install dependencies
pip install -r requirements.txt

# Run the application
streamlit run app.py
```

📋 Module Overview

1. app.py - Master Application

- **Purpose**: Main entry point that coordinates all modules
- **Features**:
  - Tab navigation between modules
  - Sidebar with system status
  - Session management
  - Error handling for all modules

2. **shared_config.py** - Configuration & Utilities

- **Purpose**: Common configuration and utility functions
- **Contains**:
  - Application configuration (field definitions, constants)
  - Database connection helpers
  - Auto-population functions
  - Session state management
  - Common utilities used across modules

3. **equipment_manager.py** - Equipment Management

- **Purpose**: Add, edit, and manage equipment data
- **Features**:
  - Smart auto-fill from database
  - Excel-like data interface
  - Complete equipment field management
  - Direct database save operations
  - Excel data import functionality

4. **search_equipment.py** - Search & Analysis

- **Purpose**: Search and analyze equipment data
- **Features**:
  - Quick search across all fields
  - Advanced search with multiple criteria
  - Data analysis and statistics
  - Export to Excel functionality
  - Equipment database overview

5. **network_visualization.py** - Network Visualization

- **Purpose**: Interactive relationship visualization
- **Features**:
  - Customer-Project-Equipment relationships
  - Circular network layouts

    - Interactive exploration tools
    - Network statistics and analysis
    - Multiple layout options (Circular, Force-Directed, Hierarchical)

🔧 Benefits of Modular Architecture

✅ Maintainability

- Each module handles a specific functionality
- Easy to update one feature without affecting others
- Clear separation of concerns

✅ Scalability

- Add new modules easily
- Extend existing modules independently
- Modular testing and debugging

✅ Code Organization

- Logical grouping of related functions
- Easier to navigate and understand
- Better collaboration between developers

✅ Performance

- Only load required modules
- Better memory management
- Faster startup times

🎯 Usage Examples

Running Individual Modules

```python
```

```
# In app.py, each module is instantiated and rendered:
equipment_manager = EquipmentManager()
equipment_manager.render()

search_equipment = SearchEquipment()
search_equipment.render()

network_viz = NetworkVisualization()
network_viz.render()
```

Accessing Shared Functions

```python
# All modules can access shared utilities:
from shared_config import (
    Config, get_user_identity, auto_populate_field,
    find_equipment_table_name, get_specification_columns
)
```

Adding New Features

```python
# To add a new module:
# 1. Create new_module.py
# 2. Import shared_config for common utilities
# 3. Add to app.py tabs
# 4. Update requirements.txt if needed
```

🔧 Configuration

Database Settings

- **TestDB**: Equipment data storage and retrieval
- **PowerApps**: Configuration and network visualization data
- **Auto-detection**: Automatically finds correct table names

Excel Integration

- **Built-in st.data_editor**: Excel-like interface (default)
- **Optional components**: streamlit-excel-table, Mito (advanced)

- **Import/Export**: Full Excel compatibility

Network Visualization

- **Optional libraries**: networkx, pyvis
- **Automatic fallback**: Installation guide if libraries missing
- **Performance tuning**: Configurable node limits

🚀 Getting Started

1. **Install dependencies**: `pip install -r requirements.txt`
2. **Run application**: `streamlit run app.py`
3. **Test connections**: Use sidebar "Test All Connections"
4. **Start with Equipment Manager**: Add your first equipment data
5. **Explore Search**: Find and analyze your equipment
6. **Visualize Networks**: See equipment relationships

📊 Monitoring and Logs

- **Application logs**: Stored in `logs/app.log`
- **Error tracking**: Each module logs errors independently
- **Performance monitoring**: Database query performance tracked
- **User activity**: Equipment changes logged with user identity

---

🎉 **Your equipment management system is now fully modular, maintainable, and scalable!**