CSV Loading Integration Guide

Overview

Your AAR system already has the CSV loading functionality implemented in the `data_management_tab.py` file. This guide will help you restore and enhance this functionality.

Quick Solution

1. Replace the Data Management Tab

Replace your existing `ui/components/data_management_tab.py` with the enhanced version I've provided. This version includes:

- **Enhanced file browsing** with better user experience
- **Improved data preview** with tabbed interface
- **Robust error handling** and progress indication
- **Sample data generation** for testing
- **Column selection** with domain-specific auto-selection
- **Data validation** capabilities
- **Export functionality** for processed data

2. Update Your Main Application

Ensure your `main_application.py` properly imports and uses the data management tab:

```python
# In _create_tabs() method
try:
    from ui.components.data_management_tab import DataManagementTab
    self.data_tab = DataManagementTab(self.notebook, self.event_bus, self)
    self.notebook.add(self.data_tab.frame, text="Data Management")
    self.logger.info("Data Management tab created successfully")
except Exception as e:
    self.logger.error(f"Error creating Data Management tab: {e}")
    self._create_placeholder_tab("Data Management")
```

3. Test the Functionality

Use the test script I provided to verify the CSV loading works:

```bash
python test_csv_loading.py
```

Key Features Restored

🗂️ File Selection

- **Browse Button**: Select CSV or Excel files
- **File Path Display**: Shows selected file with size information
- **Load Button**: Loads the selected file with progress indication
- **Sample Data**: Generate demonstration data for testing

📊 Data Preview

- **Table View**: Preview loaded data in a scrollable table
- **Data Information**: Detailed column and quality information
- **Progress Indication**: Real-time loading progress

🎯 Column Selection

- **Available Columns**: Shows all columns in the dataset
- **Selected Columns**: Columns chosen for analysis
- **Domain-Based Auto-Selection**: Automatically selects required columns
- **Bulk Operations**: Add/remove all columns at once

⚙️ Analysis Configuration

- **Domain Selection**: Choose analysis domain (Safety, Network, etc.)
- **Analysis Level**: Individual, Squad, or Platoon
- **Priority Settings**: Normal, High, or Low priority
- **Validation Options**: Enable/disable data quality checks

🚀 Analysis Execution

- **Start Analysis**: Begin processing with selected configuration
- **Data Validation**: Check data quality before analysis
- **Export Options**: Save processed data in various formats
- **Progress Monitoring**: Track analysis progress

Integration Steps

Step 1: File Replacement

1. Back up your current `data_management_tab.py`
2. Replace it with the enhanced version
3. Ensure all imports are available

Step 2: Dependencies Check

Verify these modules are available:

- `pandas` for data processing
- `numpy` for numerical operations
- `tkinter` for GUI components
- Your custom modules (`core.event_bus`, `core.models`)

Step 3: Testing

1. Run the test script to verify functionality
2. Test with your actual CSV files
3. Verify integration with your main application

Step 4: Customization

Modify the code as needed for your specific requirements:

- Add custom validation rules
- Modify column requirements for different domains
- Adjust UI layout and styling
- Add additional export formats

Troubleshooting

Common Issues

1. **Import Errors**
    - Ensure all required modules are installed
    - Check that custom modules are in the correct path
    - Verify Python path includes your project directory

2. **File Loading Errors**
    - Check file permissions
    - Verify CSV format is valid
    - Ensure file is not locked by another application
3. **UI Display Issues**
    - Verify tkinter is properly installed
    - Check screen resolution compatibility
    - Ensure all UI elements are properly packed/gridded

Error Messages

- **"Please select a file first"**: Use browse button to select CSV
- **"Unsupported file format"**: Only CSV and Excel files are supported
- **"Error loading data"**: Check file format and permissions
- **"Please select columns"**: Choose columns in the selection area

Advanced Features

Custom Data Validation

Add custom validation rules in the `_validate_data()` method:

```python
def _validate_data(self):
    # Add your custom validation logic here
    if 'required_column' not in self.current_data.columns:
        issues.append("Required column missing")
```

Domain-Specific Column Requirements

Modify the `_get_required_columns_for_domain()` method:

```python
```

```
def _get_required_columns_for_domain(self, domain: AnalysisDomain) -> List[str]:
    domain_columns = {
        AnalysisDomain.SOLDIER_SAFETY: ['callsign', 'falldetected', 'casualtystate'],
        AnalysisDomain.NETWORK_PERFORMANCE: ['callsign', 'rssi', 'mcs'],
        # Add your custom domains here
    }
    return domain_columns.get(domain, ['callsign', 'timestamp'])
```

Custom Export Formats

Extend the export functionality:

```python
def _export_data(self):
    # Add support for additional formats
    filetypes = [
        ("CSV files", "*.csv"),
        ("Excel files", "*.xlsx"),
        ("JSON files", "*.json"),  # Add JSON support
        ("XML files", "*.xml"),    # Add XML support
    ]
```

Best Practices

1. **Data Quality**: Always validate data before analysis
2. **Error Handling**: Implement comprehensive error handling
3. **User Feedback**: Provide clear status messages and progress indication
4. **Memory Management**: Handle large files efficiently
5. **File Formats**: Support multiple input formats
6. **Column Mapping**: Provide flexible column mapping options

Next Steps

1. **Test thoroughly** with your actual data files
2. **Customize** the interface for your specific needs
3. **Add validation rules** for your data requirements
4. **Extend export options** as needed
5. **Integrate** with your analysis engines

Support

If you encounter any issues:

1. Check the error logs for detailed information
2. Verify all dependencies are properly installed
3. Test with the provided sample data first
4. Refer to the troubleshooting section above

The CSV loading functionality is now fully restored and enhanced with additional features for better user experience and data management capabilities.