Generic Transaction Analysis Tool

A scalable, adaptive Python tool for analyzing financial transaction data from any business type. Handles datasets from thousands to 1 million+ records with intelligent categorization and comprehensive reporting.

🌟 Key Features

- **Universal Compatibility**: Works with any transaction data format without pre-analysis
- **Scalable Architecture**: Handles 1M+ records efficiently with memory optimization
- **Intelligent Categorization**: Automatically learns transaction patterns and creates categories
- **Flexible Column Mapping**: Easily extensible for new file formats
- **Comprehensive Analysis**: 15+ different analytical reports and visualizations
- **Export Formats**: Excel (multi-sheet), PDF reports
- **Date Range Filtering**: Filter analysis by specific date ranges
- **Multi-Account Support**: Handles multiple accounts in single or multiple files
- **Batch Processing**: Process multiple files simultaneously

📁 Project Structure

```
funding_alt/                          # 🏢 Transaction Analysis Tool
├── requirements.txt                  # 📋 Production dependencies
├── requirements-dev.txt              # 📋 Development/docs dependencies
├── build_docs.sh                     # 🔨 Documentation build script
├── main.py                           # 🎯 Interactive guided interface
├── README.md                         # 📖 Main documentation
├──
├── 📁 docs/                          # 📚 Documentation directory
│   ├── INSTALLATION.md               # 🔧 Setup and installation guide
│   ├── PROJECT_SUMMARY.md            # 📊 Project summary
│   ├── EXCEL_EXPORT_DOCS.md          # 📊 Excel export guide
│   └── Tool_Documentation.pdf        # 📄 PDF Documentation
├──
├── 📁 inputs/                        # 📁 Sample input files
│   └── 01_sample_statement.xlsx      # 📄 Sample transaction data
├──
├── 📁 scripts/                       # 🔧 Command-line executables
│   ├── __init__.py
│   ├── transaction_analyzer.py       # 💻 Main CLI application
│   └── readme_to_pdf.py              # 📄 README to PDF converter
├──
├── 📁 src/                           # 💡 Core business logic
│   ├── __init__.py                   # 📚 Package exports
│   ├── data_loader.py                # ⚡ High-performance data loading
│   ├── categorizer.py                # 🤖 Intelligent auto-categorization
│   ├── analysis_engine.py            # 📊 Scalable analysis algorithms
│   ├── visualizer.py                 # 📈 Dynamic chart generation
│   ├── exporter.py                   # 📄 Excel export functionality
│   └── decimal_utils.py              # 🔢 Decimal precision utilities
├──
├── 📁 config/                        # ⚙️ Configuration management
│   ├── __init__.py
│   └── settings.py                   # 🔧 Configuration & mappings
├──
├── 📁 output/                        # 📤 Generated reports and charts
│   └── [timestamp_folders]/          # 📁 Organized by analysis run
│       ├── analysis.xlsx             # 📊 Main analysis report
│       ├── simple.xlsx               # 📈 Simplified report
│       ├── charts/                   # 📊 Generated visualizations
│       └── account_[num]/            # 🏦 Per-account reports
├──
├── 📁 logs/                          # 📜 Application logs
└── 📁 venv/                          # 🐍 Python virtual environment
```

🚀 Quick Start

1. Setup Environment

```
# Create virtual environment (if not exists)
python3 -m venv venv

# Activate virtual environment
source venv/bin/activate  # On macOS/Linux
# or
venv\\Scripts\\activate     # On Windows

# Install required packages (including PDF support)
pip install pandas openpyxl matplotlib seaborn psutil xlsxwriter reportlab
```

2. Usage Options (Choose Your Preferred Interface)

🎯 Interactive Guided Mode

```
# Step-by-step guided interface (perfect for beginners)
python main.py
```

Features: File selection menu, guided options, shows equivalent CLI command

⚡ Command Line Mode

```
# Basic analysis
python scripts/transaction_analyzer.py --input statements.xlsx

# With date filtering
python scripts/transaction_analyzer.py --input statements.xlsx \\
    --start-date 2024-01-01 --end-date 2024-12-31

# Filter specific accounts
python scripts/transaction_analyzer.py --input statements.xlsx \\
    --accounts 3995 5968

# Generate separate reports for each account
python scripts/transaction_analyzer.py --input statements.xlsx \\
    --separate-accounts

# Process multiple files
python scripts/transaction_analyzer.py --input file1.xlsx file2.xlsx --
merge
```

📊 Expected Input Format

The tool automatically detects columns but expects these standard column types:

Required Columns:

- Account Number: `AccNo`
- Transaction Date: `Date`
- Description: `MainDesc`
- Amount: `Amount`

Adding New Column Formats

To support new file formats, simply update the arrays in `config/settings.py`:

```
COLUMN_MAPPINGS = {
    'account_number': ['AccNo', 'Account_Number', 'AccountID'],  # Add new
variations here
    'transaction_date': ['Date', 'Transaction_Date', 'Txn_Date'],  # Add
new variations here
    # ... etc
}
```

🎯 Generated Analysis Reports

1. Basic Statistics

- Dataset overview (transactions, accounts, merchants, date ranges)
- Financial summary (total income, expenses, net position)
- Transaction distribution analysis

2. Account Analysis

- Per-account financial metrics
- Account activity status (active/inactive)
- Account comparison (if multiple accounts)

3. Merchant Analysis

- Top merchants by frequency and amount
- Revenue source identification
- Expense vendor analysis

4. Intelligent Categorization

- Automatic transaction categorization based on patterns
- Similar merchant grouping
- Category performance metrics
- Exportable category mappings for manual review

5. Temporal Analysis

- Monthly/weekly transaction trends

- Seasonal patterns
- Weekday vs weekend analysis
- Growth rate calculations

6. Cash Flow Analysis

- Income vs expense tracking
- Monthly cash flow trends
- Net position analysis
- Balance validation (if balance column provided)

7. Outlier Detection

- Statistical outlier identification
- Large transaction flagging
- Potential duplicate detection

8. Data Quality Analysis

- Missing data reporting
- Data consistency checks
- Currency validation
- Potential data issues flagging

📈 Generated Visualizations

The tool automatically generates relevant charts with **context labels** for clarity:

Chart Features:

- **Context-aware titles**: Charts clearly indicate if they're for individual accounts or combined analysis
- **Smart filenames**: Include account numbers or data source names for easy identification
- **Adaptive generation**: Only creates charts relevant to your data

Available Chart Types:

- **Amount Distribution**: Histogram of transaction amounts (income vs expenses)
- **Merchant Frequency**: Top merchants by transaction count
- **Monthly Trends**: Transaction volume and amount trends over time
- **Weekday Patterns**: Day-of-week transaction analysis
- **Income vs Expenses**: Financial flow visualization and comparison
- **Balance Trends**: Account balance over time (if balance column exists)
- **Category Distribution**: Pie chart of transaction categories (if categorized)
- **Cash Flow Analysis**: Monthly income/expense flow charts
- **Transaction Outliers**: Largest transactions by amount
- **Account Comparison**: Multi-account comparison charts (if multiple accounts)

📄 Export Formats

Excel Reports

- **Comprehensive Multi-sheet Analysis** (`*_analysis.xlsx`):
  - Raw transaction data (with auto-categories)
  - Executive summary with key metrics
  - Account analysis (individual account performance)
  - Merchant analysis (top vendors by frequency and spend)
  - Category breakdown (auto-generated categories)
  - Temporal analysis (monthly trends, weekday patterns)
  - Active/Inactive accounts analysis (smart context-aware)
  - Outlier detection (unusual transactions)
  - Data quality assessment
- **Simplified Report** (`*_simple.xlsx`) - Optional:
  - Essential metrics only
  - Clean formatted summary
  - Perfect for executive reporting

📋 **Detailed Excel Documentation**: See docs/EXCEL_EXPORT_DOCUMENTATION.md for complete details about each sheet's structure, columns, and business use cases.

PDF Reports

- **Comprehensive PDF Analysis** (`*_report.pdf`) - **New Feature!**:
  - Professional executive-ready PDF reports combining tables and charts
  - **Cover page** with report metadata and key highlights
  - **Executive summary** with critical financial metrics

- **Financial overview** with embedded charts and trend analysis
- **Account performance** tables with activity status and metrics
- **Transaction analysis** with categorization breakdown
- **Visual analytics dashboard** with all generated charts embedded
- **Data quality assessment** with validation results
- Optimized for printing and executive presentation
- Charts automatically resized and embedded with proper aspect ratios

PDF Generation Requirements:

```
# PDF functionality requires reportlab
pip install reportlab
```

Enable PDF Export:

```
# Generate PDF report along with Excel
python scripts/transaction_analyzer.py --input data.xlsx --pdf-export

# PDF with custom output name
python scripts/transaction_analyzer.py --input data.xlsx --pdf-export --
output quarterly_report

# Combined Excel and PDF with all features
python scripts/transaction_analyzer.py --input data.xlsx \\
    --pdf-export --simple-export --separate-accounts
```

Separate Account Reports

- **Individual account analysis** (when `--separate-accounts` used):
  - Dedicated folder per account (`account_[number]/`)
  - Account-specific Excel reports
  - Charts with account context labels

⚙️ Configuration

Performance Settings (1M+ Records)

- **Chunk Processing**: 50K record chunks for memory efficiency
- **Data Type Optimization**: Automatic memory optimization
- **Categorical Data**: String-to-category conversion for memory savings
- **Memory Limit**: Configurable RAM usage limits

Categorization Settings

- **Minimum Frequency**: Transactions needed to create auto-category (default: 5)
- **Similarity Threshold**: String similarity for grouping merchants (default: 0.85)
- **Smart Grouping**: Automatically group similar merchant names
- **Max Categories**: Limit auto-generated categories (default: 100)

File Processing

- **Supported Formats**: `.xlsx`, `.xls`, `.csv`
- **Large File Support**: Up to 1GB files
- **Multiple Sheets**: Process all Excel sheets
- **Encoding Detection**: Auto-detect CSV encoding

🔧 Advanced Usage

Date Range Filtering

```
# Analyze last 6 months
python transaction_analyzer.py --input data.xlsx \\
    --start-date 2024-06-01 --end-date 2024-12-31

# Quarterly analysis
python transaction_analyzer.py --input data.xlsx \\
    --start-date 2024-10-01 --end-date 2024-12-31
```

Multi-File Processing

```
# Combine multiple bank statement files
python transaction_analyzer.py \\
    --input jan2024.xlsx feb2024.xlsx mar2024.xlsx \\
    --merge \\
    --output Q1_2024_analysis
```

Account-Specific Analysis

```
# Analyze only specific accounts
python transaction_analyzer.py --input data.xlsx \\
    --accounts 3995 5968 \\
    --output multi_account_analysis
```

Performance Optimization

```
# For very large datasets - skip charts for speed
python scripts/transaction_analyzer.py --input large_dataset.xlsx \\
    --skip-charts \\
    --simple-export

# Skip categorization for fastest processing
python scripts/transaction_analyzer.py --input large_dataset.xlsx \\
    --skip-categorization \\
    --skip-charts
```

🎛️ Command Line Options

Option	Description	Example
`--input` `-i`	Input file(s) (**required**)	`--input data.xlsx`
`--start-date` `-s`	Start date filter (YYYY-MM-DD)	`--start-date 2024-01-01`
`--end-date` `-e`	End date filter (YYYY-MM-DD)	`--end-date 2024-12-31`
`--accounts` `-a`	Filter specific account numbers	`--accounts 3995 5968`
`--output` `-o`	Output folder name	`--output my_analysis`
`--merge`	Merge multiple input files	`--merge`
`--separate-accounts`	Generate separate reports per account	`--separate-accounts`
`--skip-categorization`	Skip auto-categorization	`--skip-categorization`
`--skip-charts`	Skip chart generation	`--skip-charts`
`--export-categories`	Export category mappings for review	`--export-categories`
`--simple-export`	Also create simplified Excel report	`--simple-export`
`--pdf-export`	Generate comprehensive PDF report	`--pdf-export`
`--verbose` `-v`	Enable detailed logging	`--verbose`

🏗️ Architecture Design

Modular Architecture

- **Data Loader**: Flexible, high-performance data ingestion
- **Categorizer**: Pattern-learning categorization engine
- **Analysis Engine**: Scalable analytical algorithms
- **Visualizer**: Adaptive chart generation
- **Exporter**: Multi-format report generation

Scalability Features

- **Memory Optimization**: Automatic data type optimization
- **Chunk Processing**: Handle files larger than available RAM
- **Lazy Loading**: Load only required data sections
- **Caching**: Cache intermediate results for performance
- **Progress Tracking**: Real-time processing updates

📋 Requirements

System Requirements

- Python 3.8+
- 4GB+ RAM (recommended for 1M+ records)
- 1GB+ free disk space (for large outputs)

Python Dependencies

```
pandas>=1.5.0
openpyxl>=3.0.0
matplotlib>=3.5.0
seaborn>=0.11.0
psutil>=5.8.0
xlsxwriter>=3.0.0
reportlab>=4.0.0     # Required for PDF export functionality
```

✅ Testing

Test with sample data:

```
# Basic test
python transaction_analyzer.py --input inputs/01_sample_statement.xlsx

# Full feature test
python transaction_analyzer.py --input inputs/01_sample_statement.xlsx \\
    --start-date 2024-10-01 \\
    --export-categories \\
    --simple-export \\
    --verbose
```

🤝 Contributing

To extend the system for new file formats:

1. Update column mappings in `config/settings.py`
2. Test with new file format
3. Update documentation

To add new analysis types:

1. Add analysis method to `src/analysis_engine.py`
2. Add corresponding visualization in `src/visualizer.py`
3. Update export functionality in `src/exporter.py`

📚 Documentation

Complete documentation is available in the `docs/` directory:

- **Installation Guide** - Detailed setup instructions and troubleshooting
- **Project Summary** - Project overview, achievements, and technical details
- **Excel Export Documentation** - Complete guide to Excel file structure and contents
- **PDF Documentation** - Professional PDF version of this README

🔨 Building Documentation

To regenerate the PDF documentation from README.md:

```
# Quick build (installs dependencies automatically)
./build_docs.sh

# Manual build
pip install -r requirements-dev.txt
python scripts/readme_to_pdf.py
```

Generated PDF Features: - Professional formatting with proper typography - Automatic page breaks and section organization - Code syntax highlighting and table formatting - Page numbers and generation timestamps - Print-ready layout optimized for A4 paper

📞 Support

For questions or issues:

1. Check the generated log files in the `logs/` directory
2. Run with `--verbose` flag for detailed debugging
3. Review the data quality report in the Excel output
4. Consult the detailed documentation in the `docs/` directory
