Ticket Management System

📋 Overview

The Ticket Management System is a comprehensive Pythonbased application designed to streamline ticket management, analysis, and reporting. Built with a modern, minimalistic UI, it allows organizations to load, filter, analyze, and generate professional PDF reports from ticket data.

Version: 1.2

Created by: Thomas Ragen

Last Updated: November 19, 2025

✨ Key Features

📊 Data Management

- Load Excel Files: Import ticket data from Excel spreadsheets with automatic column detection
- Smart Column Detection: Autodetects Category, Subcategory, Requester, and Technician columns
- Data Validation: Handles various data formats and edge cases gracefully

🔍 Advanced Filtering

- Category Filter: Filter tickets by primary category
- Subcategory Filter: Dynamically shows subcategories only after category selection
- Requester Filter: View tickets submitted by specific requesters
- Technician Filter: View tickets worked on by specific technicians
- MultiFilter Support: Combine multiple filters for precise data views
- Clear Filters: Reset all filters with one click

📈 Data Visualization

- Dynamic Pie Charts:
    1. Category distribution when viewing all categories
    2. Subcategory breakdown when viewing specific categories
    3. Realtime updates based on all active filters
- Ticket Breakdown Table: Detailed view of filtered tickets with all attributes
- Interactive Charts: Hover for details, intuitive visual representation

📄 Report Generation

- Main PDF Report: Comprehensive report with ticket breakdown table and category distribution chart
- 52 Category Specific Reports: Generate specialized PDFs for individual ticket categories:
    1. ACS Service/Repair
    2. Antivirus
    3. BTS
    4. Channels
    5. Computer Stationery
    6. CRM
    7. Data Center
    8. Databases
    9. Desktop Hardware/Software
    10. Email
    11. ESB
    12. EOD
    13. Equinox
    14. General Repairs
    15. Generator
    16. Helpdesk
    17. iApply
    18. Information Provision
    19. Internet
    20. Intranet/Corporate Website
    21. iProfits
    22. IPRS
    23. JIRA
    24. KPrinter
    25. KPLC

26. Licences
27. Lift
28. Market Place/Lead Management
29. MIPS
30. Money Gram
31. Network
32. Notifications/Alerts
33. Online Bid Bond
34. Printed Stationery
35. Printers
36. Production Scheduler
37. Profits
38. Records & Case Management
39. Reports
40. SAP BI
41. SAP Fi
42. SWIFT
43. Taxis/Cabs
44. Team Mate
45. Telephony
46. Test Environment
47. UPS
48. Vehicle
49. Vendor Support
50. Water
51. Western Union
52. Zendesk

- Requester Analysis PDF: Detailed breakdown of tickets by requester with category distribution
- Technician Analysis PDF: Detailed breakdown of tickets by technician with work distribution
- Batch Report Generation: Select multiple categories and generate all reports at once
- Smart Report Handling: Automatically skips empty categories and notifies user

🎨 Modern UI

- Minimalistic Design: Clean, professional interface with intuitive navigation
- Status Bar: Realtime feedback on all operations
- TwoColumn Layout:
    1. Left: Ticket breakdown table
    2. Right: Dynamic pie chart
- Professional Styling: ttkbootstrap "litera" theme for modern appearance
- Responsive Design: Adapts to different screen sizes (default 1400x800)
- Creator Attribution: "Created by Thomas Ragen" displayed in header
- Version Control: Version 1.2 displayed for tracking

🚀 Getting Started

Prerequisites

1. Python 3.8+ (tested on Python 3.11)
2. pip (Python package manager)
3. Windows Operating System (primary development platform)

Installation

1. Clone the Repository

```bash
git clone https://github.com/lord-ragen/Ticket-Management-System.git
cd TicketManagementSystem
```

2. Create Virtual Environment (Recommended)

```
python -m venv venv.venv\Scripts\Activate.ps1
```

3. Install Dependencies

```
pip install -r requirements.txt
```

Or manually install required packages:

```
pip install pandas openpyxl ttkbootstrap matplotlib reportlab
```

Running the Application

1. From Command Line

python ticket_app.py

2. From PowerShell (with virtual environment activated)

.\.venv\Scripts\Activate.ps1python ticket_app.py

3. From VS Code

Open the project folder

Install Python extension

Select Python interpreter

Click "Run" or press F5

📖 Usage Guide

Basic Workflow

Step 1: Load Ticket Data

Click "📂 Load Excel File" button

Select your Excel file containing ticket data

System auto-detects columns and loads data

Status bar shows: "✓ Loaded X tickets from Y categories"

Step 2: Filter Data (Optional)

Select Category: Choose from dropdown (or leave as "All Categories")

Subcategory dropdown activates only after category selection

Select Subcategory: Choose specific subcategory (shows only relevant options)

Select Requester: Filter by who submitted the ticket

Select Technician: Filter by who worked on the ticket

Chart and table update in real-time

Step 3: View Analysis

Ticket Breakdown Table (Left): Shows filtered ticket details

Category Distribution Chart (Right):

Shows category percentages (when "All Categories" selected)

Shows subcategory breakdown (when specific category selected)

Updates based on all active filters

Step 4: Generate Reports

1. Main PDF Report

Click "📊 Export PDF"

Select save location

Report includes table + chart of current view

2. Category-Specific Reports

Click "📄 Category Reports"

Check boxes for desired categories (can select multiple)

Click "Generate Selected Reports"

Choose save location

All selected reports generate as separate PDFs

Status shows: "Generated: 5 | Skipped: 2 | Failed: 0"

3. Requester Analysis

Click "📈 Requester Stats"

Select save location

PDF shows tickets by requester with category breakdown

4. Technician Analysis

Click "👨‍🔧 Technician Stats"

Select save location

PDF shows tickets by technician with work distribution

Step 5: Clear and Reset

Click "🔄 Clear Filters" to reset all filters

Returns to viewing all tickets

📊 Excel File Format

- Required Columns

Your Excel file should contain the following columns:

Column Name	Type	Description
Category	String	Main ticket category (e.g., "Email", "Network", "CRM")
Subcategory	String	Sub-classification (e.g., "Distribution List", "Connectivity")
Subject	String	Ticket title/description
Description	String	Detailed issue description
Status	String	Current status (Open, In Progress, Closed, etc.)
Priority	String	Priority level (High, Medium, Low)
Date Raised	Date	When ticket was created
Requester	String	Name of person who requested the ticket
Technician / Assigned To	String	Name of person working on the ticket

- Column Detection

The system automatically detects:

1. **Category Column**: Looks for "Category", "Service", "System"
2. **Subcategory Column**: Looks for "Subcategory", "Sub-Category", "Type"
3. **Requester Column**: Looks for "Requester", "Submitted By", "Reporter"
4. **Technician Column**: Looks for "Technician", "Assigned To", "Assigned By"

If your columns use different names, the system will attempt to auto-detect based on content patterns.

🎯 Features in Detail

- Dynamic Filtering

The filtering system works intelligently:

1. Category-First Approach

Select a category first

Subcategory dropdown only shows subcategories in that category

Prevents invalid filter combinations

2. Independent Filters

Requester and Technician filters work independently

Can be combined with Category filters

All combinations update chart and table in real-time

3. Multi-Select Support

Multiple categories can be selected by using combobox dropdown

Multiple requesters/technicians shown in filtered results

Table shows all matching records

- Intelligent Report Generation
    1. Empty Category Handling

If category has no tickets: Report is skipped

User is notified: "Category: No tickets raised"

Final summary shows skipped reports

2. Batch Processing

Generate multiple category reports at once

Real-time status updates during generation

Individual error handling for each category

Summary report at completion

3. PDF Features

Professional formatting with ReportLab

Category-specific charts and statistics

Consistent branding and layout

Supports special characters and Unicode

- Chart Intelligence

The pie chart automatically adapts:

1. All Categories View

Shows: Email (35%), Network (25%), CRM (20%), Other (20%)Title: Category Distribution (50 0 tickets)

2. Specific Category View

Shows: Outlook (40%), Distribution List (30%), Setup (30%)Title: Subcategory Breakdown - E mail (175 tickets)

3. With Active Filters

If Requester="John Doe" selected:Shows only tickets from John DoeAll charts/tables update acc ordingly

📂 Project Structure

- Ticket-Management-System/

├── [ticket_app.py](http://_vscodecontentref_/0) # Main application file

├── ticket_tagger_*.py # 47 category-specific tagger modules

│ ├── [ticket_tagger_acservicerepair.py](http://_vscodecontentref_/1)

│ ├── [ticket_tagger_antivirus.py](http://_vscodecontentref_/2)

│ ├── [ticket_tagger_bts.py](http://_vscodecontentref_/3)

│ ├── ... (45 more files)

│ └── [ticket_tagger_zendesk.py](http://_vscodecontentref_/4)

├── [requirements.txt](http://_vscodecontentref_/5) # Python dependencies

├── README.md # This file

└── .gitignore # Git ignore rules

- Tagger Modules

Each ticket_tagger_*.py file contains:

1. tag_ticket(): Analyzes ticket data and applies category-specific tags
2. generate_*_pdf(): Creates PDF report with charts and statistics
3. Regex patterns for intelligent ticket classification
4. ReportLab formatting for professional PDFs

🔩 Configuration

- Modify Themes

Edit line in ticket_app.py:

```
style = Style(theme="litera")  # Change to: "darkly", "flatly", "journal", etc.
```

- Change Window Size

Edit line in ticket_app.py:

```
root.geometry("1400x800")  # Change dimensions as needed
```

- Add New Categories

Create new file: ticket_tagger_newcategory.py

Add to imports in ticket_app.py

Update category list in application

Add tagging logic to detect category

- Customize Column Names

In load_excel() method, modify detection logic:

Look for custom column namesif 'Your Column' in columns: self.category_col = 'Your Colu
mn'

🐛 Troubleshooting

Error: "~~ImportError: cannot import name 'generate_*_pdf'~~"

Cause: Tagger module not found or function name mismatch

Solution: Ensure tagger file exists and function name matches import statement

Error: "~~No module named 'pandas'~~"

Cause: Dependencies not installed

Solution: pip install -r requirements.txt

Error: ~~Empty Pie Chart~~

Cause: No data matches current filters

Solution: Adjust filters to be less restrictive

Load Excel file with more tickets

Clear filters to reset

Error: ~~PDF Generation Fails~~

Cause: Invalid save location or file permissions

Solution: Ensure folder has write permissions

Check disk space is available

Use standard file paths (not network drives)

Error: ~~Cannot Load Excel File~~

Cause: File format unsupported or corrupted

Solution: Use .xlsx format (not .xls)

Verify file is not corrupted

Check column headers are present

Ensure no merged cells in data

📊 Data Analysis Examples

Example 1: Email Category Analysis

Load ticket data

Select Category: "Email"

Pie chart shows: Outlook (45%), Distribution List (35%), Setup (20%)

Table shows all Email tickets with details

Export PDF to share analysis

Example 2: Technician Performance

Load ticket data

Click "👨‍💼 Technician Stats"

PDF shows tickets handled by each technician by category

Identify high-performer and workload distribution

Example 3: Requester Activity

Load ticket data

Click "📈 Requester Stats"

PDF shows tickets submitted by each requester

Identify frequent requesters and their needs

Example 4: Category Trend Analysis

Load ticket data

Select each category one by one

Note subcategory distribution

Generate reports to compare categories

Identify trends and problem areas

🔐 Security & Best Practices

- Data Privacy

No data is sent to external servers

All processing happens locally

Excel files remain on your machine

Generated PDFs are saved only where you specify

- File Handling

Always backup original Excel files

Use read-only mode for data sources

Store PDFs in secure locations

Regular cleanup of temporary files

- Performance

Recommended max file size: 50,000 tickets

For larger files, split into multiple smaller files

Use filters to reduce processing load

Clear filters after viewing for fresh state

🚀 Advanced Usage

- Batch Processing Multiple Files

Create a folder with multiple Excel files

Load first file

Generate reports

Repeat for other files

Compare generated reports

- Custom Analysis Workflow

Load data

Filter by Requester

Filter by Category

Generate category-specific PDF

Use data for resource allocation

- Performance Optimization

For large datasets:

1. Use filters to reduce data size
2. Generate reports for specific categories
3. Clear filters between analyses
4. Close and reopen app for fresh state

🤝 Contributing

To contribute to this project:

1. Fork the repository
2. Create a feature branch (git checkout -b feature/amazing-feature)
3. Make your changes
4. Commit your changes (git commit -m 'Add amazing feature')
5. Push to the branch (git push origin feature/amazing-feature)
6. Open a Pull Request

Guidelines

1. Follow PEP 8 coding standards
2. Add comments for complex logic
3. Update README for new features
4. Test thoroughly before submitting

📧 **Support & Contact**

Project Creator: Thomas Ragen

Email: thomas95ragen@gmail.com

Version: 1.2

Last Updated: November 19, 2025

For support, issues, or feature requests:

1. Create an issue in the repository
2. Contact the project maintainer
3. Check existing issues for solutions

✅ Changelog

Version 1.2 (November 19, 2025)

- ✨ Added dynamic pie chart filtering
- ✨ Implemented linked category-subcategory filters
- ✨ Added Requester and Technician filters
- ✨ Added Technician Stats PDF generation
- ✨ Modernized UI with minimalistic design
- ✨ Added 47 category-specific tagger modules
- ✨ Implemented real-time status bar updates
- 🐛 Fixed empty category report handling
- 🐛 Improved error notifications
- 📚 Added comprehensive documentation

Version 1.1

- Added batch report generation
- Improved Excel column detection
- Enhanced PDF formatting

Version 1.0

- Initial release
- Basic ticket loading and filtering
- PDF report generation

🙏 **Acknowledgments**

- ttkbootstrap for modern UI theme
- ReportLab for PDF generation
- Pandas for data manipulation
- Matplotlib for charting
- All users and contributors

Thank you for using Ticket Management System! 🎉

For the best experience:

1. Keep your Excel files organized
2. Use consistent category names
3. Generate reports regularly
4. Share insights with your team
5. Provide feedback for improvements