📁 Master Project State: Skeptic Analyst Agent

Date: Saturday, December 6, 2025
Status: Part I & Part II Complete ✅ | Ready for Part III 🚀

---

1. Project Identity

- **Name:** Skeptic Analyst Agent
- **Goal:** An AI Agent that audits any CSV data, cleans it interactively ("The Surgeon"), and eventually transforms/visualizes it.
- **Persona:** Paranoid, distrustful auditor who refuses to process bad data.
- **Tech Stack:** Python, LangChain, OpenAI (GPT-4o), Polars, ReportLab.

---

2. Current Capabilities (What Works)

We have successfully built a **Universal Data Surgeon** that is dataset-agnostic.

A. 🧠 Universal Audit (`audit_tools.py`)

Type-Based Logic: No longer looks for hardcoded columns like "Sales" or "Region".

Checks:

- **Negatives:** Scans all numeric columns for negative values
- **Outliers:** Uses IQR (Interquartile Range) on all numeric columns (threshold adjusted for small datasets)
- **Nulls:** Checks all columns (handling both `Null` and empty strings `""`)
- **Duplicates:** Detects exact duplicate rows
- **Schema Drift:** Validates expected columns exist

Key Functions:

```python
```

```
def run_all_checks(df):
    - check_structure(df)    # Schema validation
    - check_integrity(df)    # Nulls & duplicates
    - check_validity(df)     # Range violations & outliers
```

B. 🔧 Interactive Cleaning (`cleaning_tools.py`)

Dynamic Menu: The Agent generates a custom menu based on the specific errors found in the current file.

Strategies Available:

- **Nulls:** Mean, Median, Mode, Zero, Drop Rows
- **Outliers:** Cap at Threshold, Remove Rows, Replace with Median
- **Negatives:** Make Positive, Replace with 0, Remove Rows
- **Regions/Categories:** Replace with Mode, Replace with "Unknown", Remove Rows
- **Duplicates:** Remove (keep first occurrence)

Smart Features:

- **Whitespace Stripping:** Auto-cleans on load to detect hidden duplicates
- **Undo Stack:** Revert any cleaning action (`history_stack`)
- **Auto-Pilot Mode:** Applies conservative defaults to all issues
- **Smart Filename:** Saves cleaned files as `clean_{original_name}.csv`

Key Methods:

```
python

session.load_frame(df)              # Load data with auto-cleanup
session.analyze_options()           # Generate dynamic fix menu
session.apply_fix(option_id, strat) # Apply specific fix
session.undo()                      # Revert last change
session.export_cleaned_data()       # Save to CSV
session.get_summary()               # Show current state
```

C. 🔄 The Session Loop (`app.py`)

Startup Flow:

1. Automatically scans folder for all CSV files
2. Lists files with numbered menu
3. User selects which file to load

Workflow:

```
Load File → Audit → Clean → Undo (if needed) → Export
            ↓
Type 'done' or 'switch' → Return to File Selector
```

Available Tools:

- `run_deep_audit` - Validates data against rules
- `check_cleaning_options` - Shows dynamic menu of fixes
- `apply_cleaning_fix` - Applies fix with strategy (fuzzy matching enabled)
- `undo_last_fix` - Reverts last cleaning action
- `export_cleaned_data` - Saves current state to CSV
- `generate_pdf` - Creates PDF report of audit
- `email_report` - Sends report via email (simulated)
- `get_data_summary` - Shows row/column/null/duplicate counts

Agent Features:

- **Memory:** Uses `ConversationBufferMemory` for context retention
- **Fuzzy Matching:** Maps "median" → "replace with median" automatically
- **Error Recovery:** Graceful degradation on tool failures
- **Auto-Save:** Saves after every cleaning operation

---

3. Recent Improvements (Dec 6, 2025)

✅ Completed Today:

1. **Enhanced `get_summary()`**
   - Now shows: Rows | Columns | Nulls | Duplicates
   - Added null-check for safety
2. **Error Handling in `apply_cleaning_fix()`**
   - Added try-except around summary generation
   - Prevents crashes if summary fails
3. **Auto-Pilot Success Validation**
   - Now checks if each fix succeeds before reporting
   - Accurate status messages (✓ vs ✗)
4. **Region Fix Validation**
   - Added checks for empty valid regions
   - Prevents mode calculation on empty data
5. **Fuzzy Strategy Matching**
   - "median" → "replace with median"
   - "cap" → "cap at threshold"
   - "remove" → "remove rows"

---

4. File Structure

```
skeptic-analyst-agent/
├── app.py                  # Main entry point, outer/inner loops
├── cleaning_tools.py       # CleaningSession class, fix logic
├── audit_tools.py          # Universal validation rules
├── reporting_tools.py      # PDF generation, email simulation
├── instructions.txt        # Agent persona & behavior rules
├── requirements.txt        # Dependencies
├── .env                    # OpenAI API key
├── sales_data.csv          # Test dataset (dirty)
├── clean_sales_data.csv    # Output (cleaned)
├── Audit_Report.pdf      # Generated report
└── temp_audit_log.txt    # Temporary audit results
```

5. Key Code Snippets (For Quick Reference)

Example: Running the Agent

```python
# Terminal:
python app.py

# Output:
🔍 SCANNING FOR DATASETS...
Found 2 CSV files:
  1. sales_data.csv
  2. patients.csv
Select a file to load (1-2): 1

🤖 SKEPTIC AGENT ONLINE. Type 'exit' to quit.
```

Example: Cleaning Flow

```
User: audit
Agent: ❌ AUDIT FAILED:
    - NULLS: Column 'sales' has 2 missing values
    - OUTLIERS: Found 1 sales records above 10,000

    ⚠️ DATA INTEGRITY COMPROMISED. Your options:
    1. 📄 Download PDF Report
    2. 📧 Email PDF Report
    3. 🔧 Interactive Data Cleaning

User: 3
Agent: 🔧 Available Cleaning Options:
    0. Apply ALL Recommended Fixes (Auto-Pilot)
    1. Fix 2 Nulls in 'sales' (Strategies: mean, median, mode, zero)
    2. Fix 1 Outliers (>24800) (Strategies: cap at threshold, remove rows, replace with median)

User: 0
Agent: ✅ Auto-pilot complete:
     ✓ Fixed Nulls in sales (median)
```

```
  ✓ Capped Outliers

(Current Data: Rows: 10 | Columns: 3 | Nulls: 0 | Duplicates: 0)
```

---

6. Future Roadmap (The Next Steps)

We are currently paused at the **Fork in the Road**. The Agent successfully cleans data. Now it needs to **use it**.

🔜 Part III: The "Dual-Path" Upgrade

Option 1: The Data Engineer (ETL Mode)

Goal: Convert flat CSVs into a Star Schema and load into SQL DB

Logic:

- **Auto-Classification:** Use heuristics (unique value counts) to guess Dimensions vs. Facts
- **Transformation:** Explode `clean_data.csv` into `dim_table.csv` and `fact_table.csv`
- **Loading:** Insert into local SQLite/DuckDB database

New File: `engineering_tools.py`

Key Functions to Build:

```python
def detect_dimensions(df):
    """Uses cardinality ratio to identify dimension columns"""
    # If unique_count / total_rows < 0.5 → likely dimension
    pass

def create_star_schema(df):
    """Splits data into fact and dimension tables"""
    # Returns: dict with 'fact_table' and 'dim_tables'
    pass

def load_to_database(tables, db_path):
    """Inserts tables into SQLite/DuckDB"""
    pass
```

Option 2: The Data Analyst (BI Mode)

Goal: Generate insights and stories

Logic:

- **Visualization:** Generate Python code to plot charts (Bar, Line, Scatter)
- **Storytelling:** LLM analyzes charts and writes business summary

New File: `analytics_tools.py`

Key Functions to Build:

```python
def generate_visualizations(df):
    """Creates matplotlib/plotly charts based on data types"""
    pass

def analyze_trends(df):
    """LLM analyzes data and generates insights"""
    pass

def create_dashboard(df):
    """Combines charts + narrative into HTML report"""
    pass
```

---

7. Immediate Next Action

When you restart tomorrow, paste this summary and say:

> **"Let's start building Option 1 (The Data Engineer). Create `engineering_tools.py` to auto-detect Dimensions and Facts."**

---

8. Git & GitHub Status

- **Repository:** `skeptic-analyst-agent`
- **Last Commit:** "Feature: Enhanced error handling and summary feedback"
- **Branch:** `main`

- **Uncommitted Changes:** Recent improvements to `cleaning_tools.py` and `app.py`

Suggested Next Commit Message:

```
"Refactor: Improved auto-pilot validation and error recovery"
```

---

9. Testing Checklist

Before moving to Part III, verify:

☑ Multi-file CSV selection works

☑ Audit detects all error types

☑ Cleaning menu shows correct options

☑ Auto-pilot applies all fixes correctly

☑ Undo reverts changes properly

☑ Export saves to correct filename

☑ PDF generation works

☑ Agent maintains conversation context

☑ Error handling prevents crashes

☑ Summary shows accurate stats

---

10. Known Limitations (To Address Later)

1. **Email is simulated** - Real SMTP integration needed
2. **PDF is basic** - Could use better formatting
3. **Single-user only** - No multi-session support
4. **No data validation rules config** - Hardcoded in audit_tools.py
5. **Limited to CSV** - Could support Excel, Parquet, JSON

🎉 Conclusion

What You Built Today:

- ✅ Universal data auditor (works on ANY CSV)

- ✅ Interactive cleaning with undo support
- ✅ Smart auto-pilot with conservative defaults
- ✅ Robust error handling and validation
- ✅ Professional agent persona
- ✅ Clean, modular architecture

You have built a very solid foundation. Rest well! See you tomorrow for Part III. 🚀