Here is the refined Software Development Specification for the **PriceReversalNewsSummary** application, optimized for development using the **Antigravity** framework and **Google Gemini**.

Software Design Specification: PriceReversalNewsSummary

1. Executive Summary

The **PriceReversalNewsSummary** application is a Python-based tool designed to analyze S&P 500 pricing data to identify potential price reversals. It automates the ingestion of financial data, selects specific company subsets (Big Movers, Tech Darlings, Financials), aggregates relevant news via **NewsAPI**, and leverages **Google Gemini** to generate comprehensive 3-5 page narrative reports[1111].

The application will be built as a web application using **Antigravity**, featuring a user interface for file uploads and job management, and a backend for data processing and API orchestration[222222222].

---

2. System Architecture

The system follows a modular architecture orchestrated by the Antigravity framework.

- **Frontend (Web UI):** Built within Antigravity, providing an interface for Excel uploads, run mode selection, and report retrieval[3].
- **Backend (API & Core):** A Python 3.10+ backend hosting API endpoints and core processing logic[4444].
- **External Integrations:**
  - **NewsAPI:** For fetching real-time financial news[55555].
  - **Google Gemini:** For normalizing company names and generating the final narrative analysis[666666666].

3. Functional Requirements

3.1 Data Ingestion

- **Excel Upload:** The system must accept an Excel file containing S&P 500 data with a defined schema (Ticker, Company Name, Sector, Predicted Return, etc.)[7].
- **Validation:** The ingestion module must validate columns and data types, rejecting files that do not match the schema[8].
- **Storage:** Parsed data is stored in memory for the session or in temporary storage (local/cloud)[9].

3.2 Subset Selection

The system supports three distinct execution modes for selecting companies[10]:

- **Big Movers:** Automatically selects the top 10 positive and top 10 negative predicted movers (20 total)[11111111].
- **Tech Darlings:** Selects companies based on a pre-defined configurable list (JSON/YAML) of high-activity tech companies[12121212].
- **Big Financials:** Selects companies based on a pre-defined configurable list of major financial institutions[13131313].

3.3 LLM Processing (Gemini)

- **Company Name Normalization:**
  - The system uses Gemini to refine company names from the subset into search-friendly query strings (e.g., handling abbreviations or brand names)[14141414141414141414].
  - Output is validated and parsed into JSON; failures default to the raw ticker/name[15].
- **Report Generation:**
  - Inputs: Subset data, News Summary, "Price Reversal Primer" document, and a base prompt[16161616].
  - **Context Window:** Uploads all documents to Gemini as context[17].
  - **Output:** Generates a structured 3-5 page report analyzing reversal prospects, risks, and catalysts, incorporating specific news items[18181818].

3.4 News Aggregation (NewsAPI)

- **Fetching:** Queries NewsAPI for each normalized company name, filtered by a configurable date range (e.g., 7-30 days) and English language[19191919].
- **Error Handling:** Implements rate limiting and backoff strategies; logs and skips companies with empty results[20].
- **Summary File:** Aggregates article titles, sources, URLs, and dates into a structured summary file for the LLM[21212121].

---

4. Project Structure (Antigravity)

The project will be hosted on GitHub (jaxdaddy/PriceReversalNewsSummary) and structured for Antigravity deployment[22222222].

Plaintext

```
/PriceReversalNewsSummary
├── /apps
│   ├── /web                # Antigravity Frontend components
│   │   ├── upload_form     # Excel upload UI [cite: 140]
│   │   ├── run_mode_selector # Job trigger UI [cite: 141]
│   │   └── report_view     # Status and download UI [cite: 142]
│   └── /api                # Backend Endpoints
│       └── main.py         # API Entrypoint [cite: 133]
├── /price_reversal_core    # Core Logic Modules [cite: 143]
│   ├── ingestion.py        # Excel validation & loading [cite: 144]
│   ├── subsets.py          # Selection logic [cite: 145]
│   ├── llm_normalizer.py   # Gemini name cleaning [cite: 146]
│   ├── news_fetcher.py     # NewsAPI integration [cite: 147]
│   └── report_generator.py # Gemini report creation [cite: 148]
├── /prompts
│   ├── base_prompt.txt     # Instructions for the LLM [cite: 151]
│   └── price_reversal_primer.md # Context document [cite: 152]
├── /config
│   ├── tech_darlings.json  # Configurable lists [cite: 154]
│   └── big_financials.json # Configurable lists [cite: 155]
├── .env.example            # Template for secrets [cite: 156]
└── README.md
```

---

5. API Specification

Endpoint	Method	Description	Inputs	Outputs
/upload-excel	POST	Uploads and parses the S&P 500 file [23]	File (Excel)	run_id, status
/run/{mode}	POST	Triggers the analysis	mode (big_movers,	Job Status

		pipeline [24]	etc.), run_id	
/report/{run_id}	GET	Retrieves the final generated report [25]	run_id	Markdown/HTML Report

---

6. Configuration & Security

- **Environment Variables:**
  - NEWSAPI_KEY: API key for NewsAPI[26].
  - GEMINI_API_KEY: API key for Google Gemini[27].
  - GEMINI_MODEL_NAME: Target model (e.g., gemini-2.0-flash)[28].
- **Security:**
  - .env files must be git-ignored[29].
  - API keys must be masked in all logs[30].
  - HTTPS-only endpoints for deployment[31].

7. Next Step

Would you like me to generate the **ingestion.py** code or the **base_prompt.txt** to get you started with the implementation?