**📂 Multi-Agent System (LangGraph) Directory Structure**

multi\_agent\_system/

│── README.md # Project overview & setup instructions

│── requirements.txt # Dependencies (langgraph, langchain, yfinance, etc.)

│── .env # API keys (OpenAI, etc.)

│── main.py # Entry point (CLI/Streamlit app)

│

├── config/ # Configuration files

│ ├── settings.py # Global configs (API keys, constants, model choice)

│ └── logging\_config.py # Logging setup

│

├── agents/ # Agent definitions

│ ├── \_\_init\_\_.py

│ ├── news\_agent.py # News & Info Researcher

│ ├── data\_agent.py # Data Researcher

│ ├── analyst\_agent.py # Data Analyst

│ └── financial\_expert\_agent.py # Financial Expert

│

├── tools/ # Custom tools and API integrations

│ ├── \_\_init\_\_.py

│ ├── search\_tool.py # DuckDuckGo wrapper

│ ├── finance\_tools.py # Yahoo Finance APIs (price, statements, info)

│ └── utils.py # Helper functions (retry, error handling)

│

├── tasks/ # Task definitions (nodes in LangGraph)

│ ├── \_\_init\_\_.py

│ ├── news\_task.py # Gather company news

│ ├── financials\_task.py # Gather financial data

│ ├── analysis\_task.py # Perform analysis

│ └── recommendation\_task.py # Generate recommendations

│

├── graphs/ # LangGraph workflow definitions

│ ├── \_\_init\_\_.py

│ ├── investment\_graph.py # Graph connecting agents/tasks

│ └── execution.py # Graph runner (kickoff, monitoring)

│

├── outputs/ # Generated results

│ ├── analysis/ # Analysis reports

│ ├── recommendations/ # Buy/Hold/Sell outputs

│ └── logs/ # Execution logs

│

├── tests/ # Unit and integration tests

│ ├── test\_agents.py

│ ├── test\_tools.py

│ ├── test\_tasks.py

│ └── test\_graph.py # End-to-end LangGraph workflow test

│

└── ui/ # Optional user interfaces

├── cli.py # CLI entrypoint

└── app.py # Streamlit/Web app for interactive use

**⚙️ Key Adaptations for LangGraph**

* **graphs/ replaces crew/** → This is where you define and wire up **nodes (tasks/agents)** and **edges (execution flow)** in LangGraph.
* **Tasks = nodes** → Each task\_\*.py file maps to a **LangGraph node**.
* **Execution flow in investment\_graph.py** → Defines how tasks connect (sequential, conditional, or parallel).
* **execution.py** handles graph kickoff, observability (callbacks, logging), and error handling.
* Modular design → You can easily add new graphs (e.g., risk\_graph.py, portfolio\_graph.py).