DAMG 7245 - Case Study 1 - Team 5

|  |  |
| --- | --- |
| **Summary** | In this codelab, we have designed an optimal data pipeline architecture for ﬁnancial news aggregator use case using one of the data pipeline pattern and have discussed the components and design process along justifying our selection |
| **URL** | [https://codelabs-preview.appspot.com/?file\_id=1p5ysK51slFwN5Jy3RqgzmJ6hx](https://codelabs-preview.appspot.com/?file_id=1p5ysK51slFwN5Jy3RqgzmJ6hxtisLiG2SzGBf233sDg&0) [tisLiG2SzGBf233sDg#0](https://codelabs-preview.appspot.com/?file_id=1p5ysK51slFwN5Jy3RqgzmJ6hxtisLiG2SzGBf233sDg&0) |
| **Category** | Application |
| **Environment** | Android, iOS |
| **Status** | In Status |
| **Feedback Link** | <https://github.com/aditya-kanala/DAMG7245_Team5_Assignment1> |
| **Author** | Aditya, Shikhar, Shubh |

## Table of Contents

PR/FAQ

Heading Subheading

Summary Paragraph Problem Paragraph Solution Paragraph Internal Quote Customer Quote Getting Started

External FAQ: What aspects of the use-case should end-users know about?

Internal FAQ (For Extended Team) Architecture Diagram

Research and Understanding Requirements Challenges

Types of Data Pipeline Patterns Fumble Data Pipeline Pattern

Data Pipeline Design Development

Architecting the Data Pipeline

Speciﬁc Needs and Characteristics of Fumble

Design Process and Decisions Assumptions Made

Use Cases to be supported by Fumble Open Questions

Future Improvements References

**Part 1**

# PR/FAQ

### Heading:

Fumble – Your Personal Finance News Buddy

### Subheading:

For the savvy individual who craves tailored finance news on the go.

### Summary Paragraph:

Today marks the launch of Fumble, a game changing finance which is set to revolutionize how you consume your financial news and designed to cater the unique interests of every individual. This dynamic platform uses AI wizardry, seamless data pipeline and comprehensive content integration to deliver personalized finance updates directly at your fingertips and revolutionize how users stay informed about the financial markets. Get ready for a smarter way to stay informed.

### Problem Paragraph:

Sick of drowning in a sea of irrelevant financial news? Fumble understands your pain. We've all been there— tired of sifting through irrelevant financial news? But fear not! Fumble is here to rescue you from the chaos and bring you the personalized and meaningful finance updates you crave, tailored just for you.

### Solution Paragraph:

Fumble simplifies your financial news journey. With a sleek integration of major news APIs, social media buzz, and innovative AI models to deliver a tailored news feed directly to your fingertips. Our robust data pipeline ensures real-time updates, efficient storage on AWS S3, and a user-friendly frontend powered by Streamlit. It's the antidote to information overload. Say goodbye to information overload and hello to Fumble's simplified and personalized news experience.

### Internal Quote*:*

*"Fumble is all about putting the user back in control of their news feed, providing relevant and meaningful content effortlessly."*

### Customer Quote*:*

*"Fumble is like having a personal finance news assistant. It gets me exactly what I need, without the noise. It's a game-changer!"*

### Getting Started:

Ready to dive into the world of personalized finance news? Getting started is a breeze! Just visit our website or download our application to embark on your journey to a smarter news experience and start customizing your finance news feed today.

## External FAQ: What aspects of the use-case should end-users know about?

#### What is Fumble?

* + Fumble is a personalized finance news aggregator designed to deliver tailored content to users based on their unique interests and preferences.

#### How does Fumble work?

* + Fumble utilizes advanced machine learning models, including OpenAI GPT-3, to understand user interests. It integrates major financial news APIs, social media sources, and web scraping to curate a personalized newsfeed.

#### What sets Fumble apart from other news platforms?

* + Fumble stands out with its AI-powered personalization, efficient data pipeline orchestrated by Mage, and seamless integration of Algolia for powerful full-text search capabilities.

#### Is my data secure with Fumble?

* + Yes, Fumble prioritizes user data security. Personal information is handled with utmost care, and the platform adheres to industry-standard security practices.

#### How can I get started with Fumble?

* + Getting started is easy! Simply visit our website, create an account, and experience the future of personalized finance news.

#### Can I customize my news preferences on Fumble?

* + Absolutely! Fumble allows users to customize their news preferences, ensuring a truly personalized news experience.

#### Is Fumble available on mobile devices?

* + Yes, Fumble is accessible on both desktop and mobile devices, providing a seamless experience across platforms.

#### Does Fumble support real-time updates?

* + Yes, Fumble delivers real-time updates, ensuring users stay informed about the latest developments in the financial world.

# Internal FAQ (For Extended Team):

#### Q: What is the high-level architecture of Fumble?

* + - A: Fumble follows a modular architecture involving data sources, Mage for transformation, AWS S3 for storage, Algolia for search, Fast API for API integration, and Streamlit for the frontend.

#### Q: How does the data pipeline work in Fumble?

* + - A: Fumble's data pipeline, orchestrated by Mage, ensures efficient data ingestion, transformation using AWS glue, and storage on AWS S3. It emphasizes scalability and reliability.

#### Q: What machine learning models are utilized in Fumble?

* + - A: Fumble leverages OpenAI GPT-3 for content generation and personalization, enhancing user engagement and providing a unique news experience.

#### Q: How is user data secured in Fumble?

* + - A: Fumble prioritizes user data security, implementing industry-standard practices to handle personal information securely and ensuring compliance with data protection standards.

#### Q: Which external APIs are integrated into Fumble?

* + - A: Fumble seamlessly integrates with major financial news APIs, social media platforms, and Algolia for powerful full-text search capabilities.

#### Q: What technology is used for the front end in Fumble?

* + - A: Streamlit is employed for front-end development, providing an intuitive and user-friendly interface to enhance the overall user experience.

#### Q: How is the data pipeline orchestrated and monitored in Fumble?

* + - A: Apache Airflow plays a pivotal role in orchestrating the data pipeline, monitoring task execution, and handling dependencies, ensuring a smooth workflow.

#### Q: How is Fumble optimized for scalability?

* + - A: Strategies for optimizing the architecture for scalability are in place, considering potential increases in data volume and user traffic to maintain optimal performance.

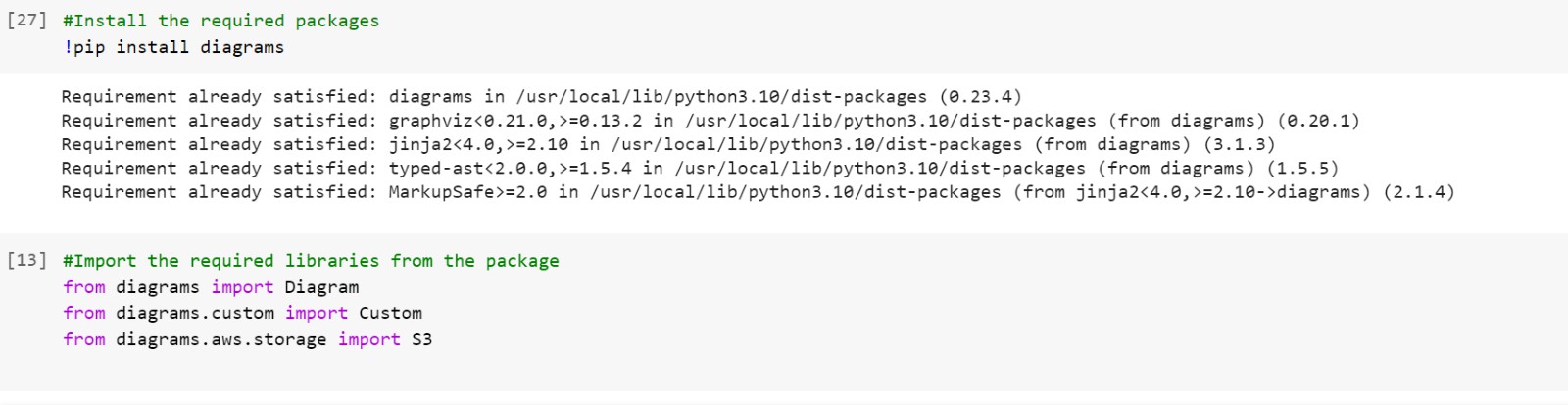
#### Q: What is the testing approach for Fumble?

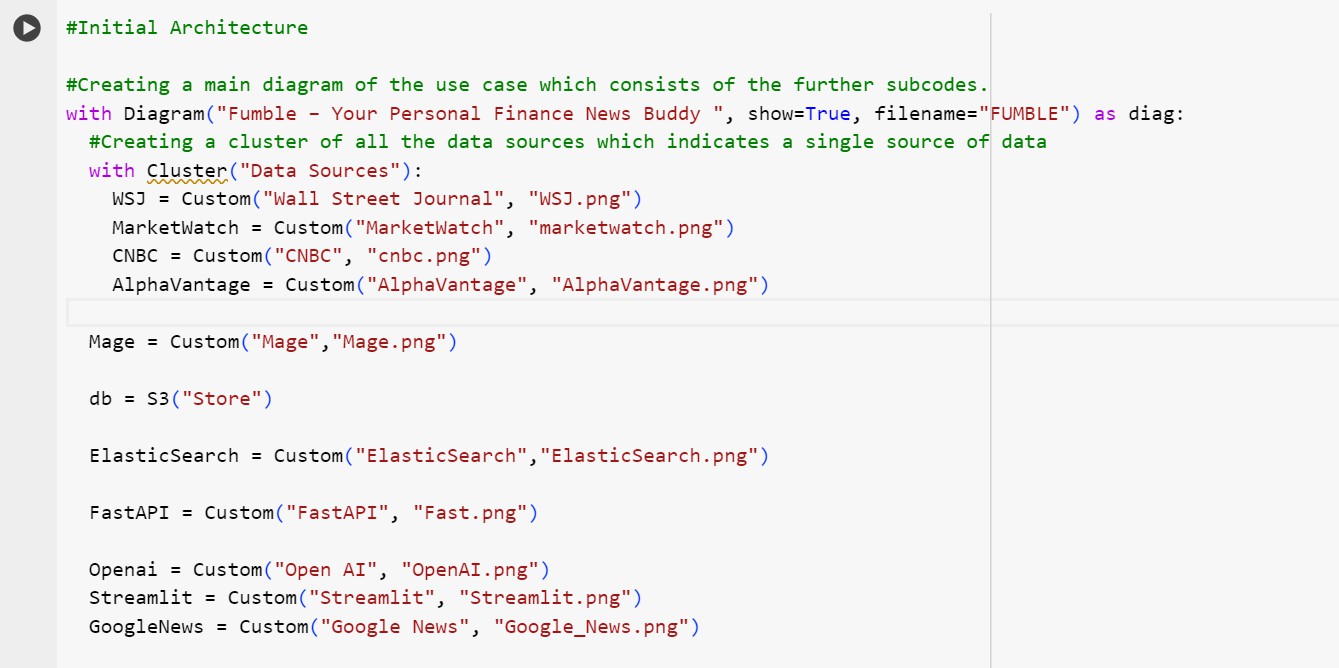
* + - A: Fumble follows a comprehensive testing approach, including unit testing, integration testing, and performance testing, to ensure the robustness and reliability of the entire system.

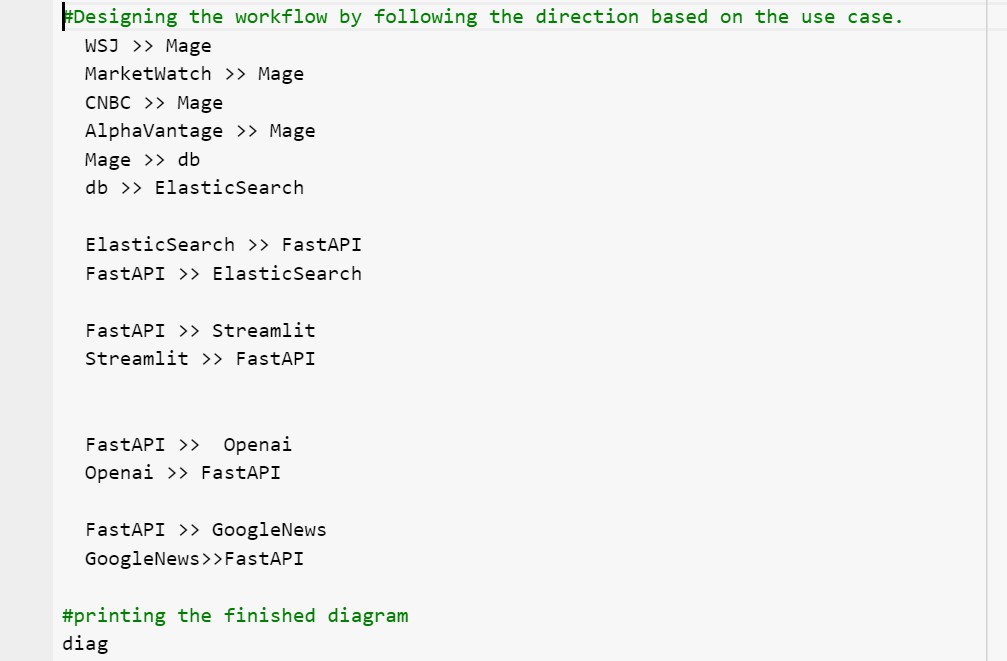
#### Q: How is knowledge transfer facilitated within the team?

* + - A: Comprehensive documentation covering the architecture, data flows, and component interactions has been created to facilitate knowledge transfer within the team, ensuring everyone is on the same page.

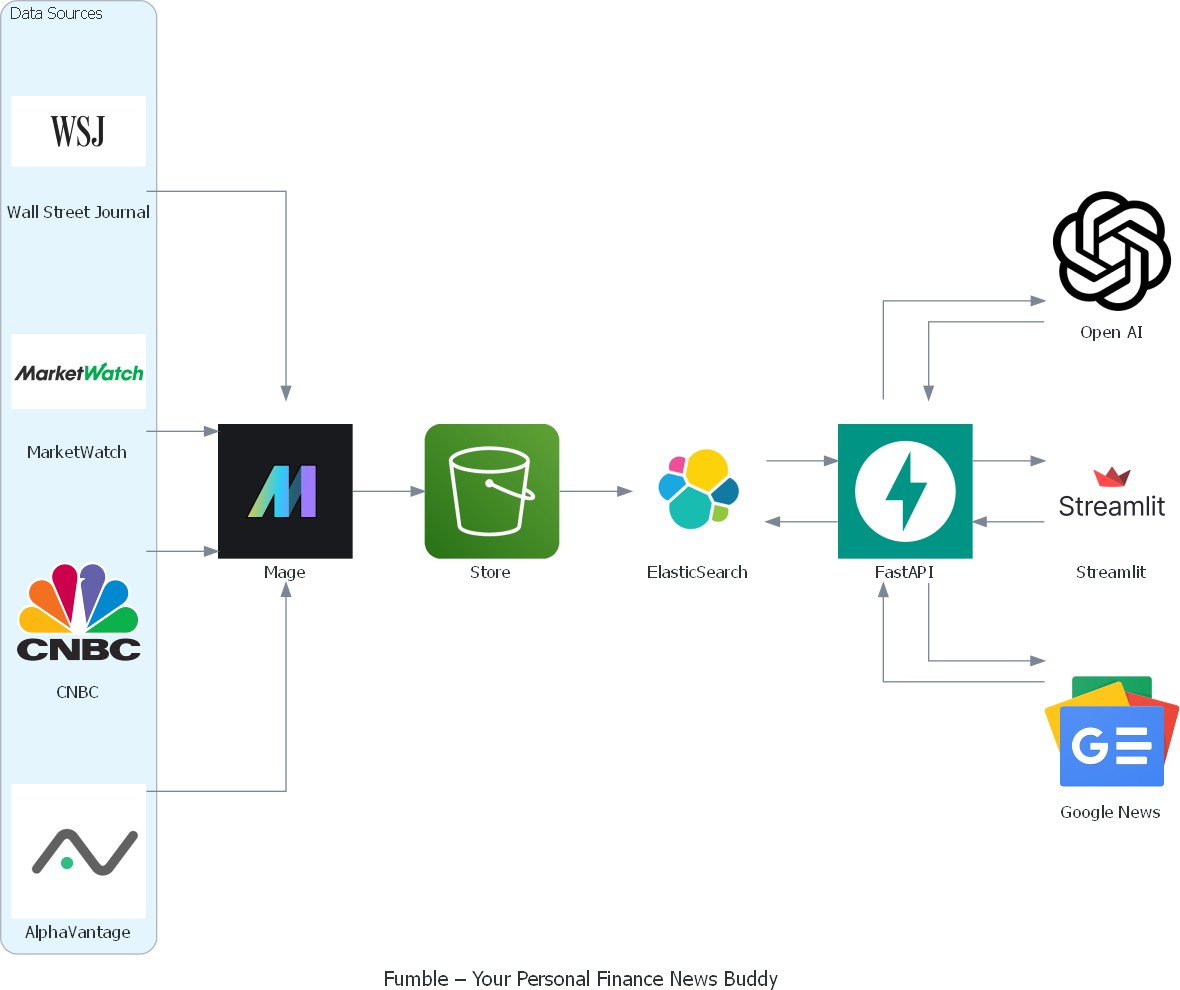
# Architecture Diagram

The architecture diagram for the use case has been designed using various libraries of the Diagram package of the python. The code and a brief explanation in the comments is given below.





The architecture diagram after successfully executing the code is as follows.



The choice of the above used architectural components is described below.

1. Data Sources:
   * Wall Street Journal (WSJ), MarketWatch, CNBC, AlphaVantage:
     + Represent various financial news and data sources.
2. Data Transformation (Mage):

* Processes and transforms data from multiple financial news sources.

1. Storage (AWS S3):

* Acts as a central repository for storing the processed data, including real-time stock price information.

1. Search and Indexing (Elasticsearch):

* Efficiently stores and indexes the transformed financial news data, enabling fast search and retrieval.

1. API Framework (Fast API):

* Exposes API endpoints for data retrieval, including live updates on stock prices and other financial news.

1. AI Models (OpenAI):

* Integrates innovative AI models to generate concise summaries of financial news articles, analyze the sentiment of news articles, identify key topics within financial news data.

1. User-Friendly Frontend (Streamlit):

* Provides a user-friendly interface for users to interact with the application, displaying personalized news content, including real-time stock prices.

1. Additional News Source (Google News):

* Serves as a crucial source for live updates on stock prices, enhancing the variety and relevance of the news feed.

The Data workflow is as follows.

1. News/financial data is scraped and sent to Mage for processing
2. Mage stores processed data into S3 object storage
3. Elasticsearch indexes the S3 data for search and analytics
4. Fast API serves API requests using Elasticsearch and OpenAI
5. Streamlit app queries FastAPI and displays summaries to end user

# Part 2 Research and Understanding

### Requirements:

To build an efficient data pipeline for the use case, Fumble as discussed earlier, it is important to understand the basic requirements of the use case being constructed.

The following are the requirements for building a data pipeline for Fumble.

* + Extract all relevant data from specified web pages and APIs without omissions.
  + Parse, clean, and normalize extracted data into structured formats with appropriate data types for storage.
  + Store accumulated records in appropriate databases in a model suitable for intended usage with sufficient capacity for growth.
  + Build a query interface and API for consumers to lookup records, filter result sets and retrieve aggregated statistics.
  + Present data extracts in consumer-friendly dashboards, reports and visualizations tailored for analysis tasks.
  + Design modular scrape architecture for high throughput, low latency data pipelines with fault tolerance.

### Challenges:

Building any data pipeline with specific requirements will always have its own challenges. The challenges which could arise while building the data pipeline for Fumble are as follows.

* + Frequent website layout/markup changes causing breaks.
  + Variety of inconsistent data formats requiring custom normalization code.
  + Planning for rapid database expansion as scrape volumes increase.
  + Slow running queries as data volumes grow larger.
  + Fitting diverse analysis needs to interface in an intuitive way.
  + Preventing bottlenecks across network, scraper, parsing and storage components.

### Types of Data Pipeline Patterns

There are few fundamental data pipeline design patterns. The given data pipeline design patterns can address the requirements and challenges as follows with each model having its own drawback:

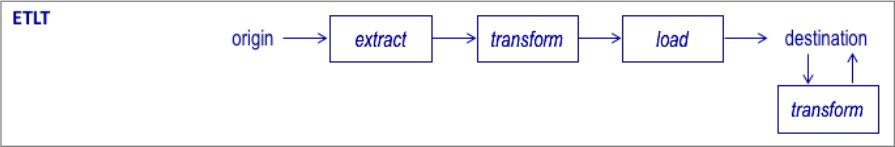
1. Raw Data Load: This pattern can help in extracting relevant data from web pages and APIs without omissions. However, the drawback is that it does not involve data transformation, which might be necessary for cleaning and normalizing the extracted data.
2. Streaming ETL: This pattern can assist in parsing, cleaning, and normalizing the extracted data into structured formats. It can also help in building a query interface and API for consumers to lookup records and retrieve aggregated statistics. However, the drawback is that it might not be suitable for rapid database expansion as scrape volumes increase.
3. ELT (Extract-Load-Transform): This pattern can help in storing accumulated records in appropriate databases and designing a modular scrape architecture for high throughput, low latency data pipelines with fault tolerance. However, the drawback is that slow running queries as data volumes grow larger might be a challenge.
4. ETLT (Extract-Transform-Load-Transform): This pattern can address the challenge of preventing bottlenecks across network, scraper, parsing, and storage components by performing initial light transformations before loading the data. However, the drawback is the approach of loading data before performing all transformations can pose challenges in terms of data governance and privacy, as sensitive data may be exposed before it is transformed.
5. Data Virtualization: This pattern can assist in presenting data extracts in consumer-friendly dashboards, reports, and visualizations tailored for analysis tasks. However, the drawback is that it might not be suitable for planning for rapid database expansion as scrape volumes increase.
6. Stream Processing: This pattern can help in addressing the challenge of frequent website layout/markup changes causing breaks. However, the drawback is that it might not be suitable for preventing bottlenecks across network, scraper, parsing, and storage components.
7. Change Data Capture (CDC): This pattern can assist in preventing bottlenecks across network, scraper, parsing, and storage components. However, the drawback is that it might not be suitable for fitting diverse analysis needs into an intuitive interface in an efficient way.

### Fumble Data Pipeline Pattern

However, considering our use case and the architecture diagram as discussed previously, the data pipeline pattern of ETLT (Extract Transform Load Transform) would be more appropriate to implement because of the reasons stated as follows:

1. Extracting all relevant data from web pages and APIs without omissions.
2. Parsing, cleaning, and normalizing data into structured formats, addressing inconsistent data formats.
3. Allowing for rapid database expansion as scrape volumes increase.
4. Building a query interface and API for efficient data retrieval.
5. Preventing bottlenecks across network, scraper, parsing, and storage components by performing initial light transformations before loading the data.

The architecture of the ETLT pipeline is as follows:



# Data Pipeline Design Development

### Architecting the Data Pipeline

The basic architecture of the use case remains the same. A few additions have been made to cluster the resources to clearly depict the type of ETLT pipeline being used.

The basic packages, libraries and the skeleton code remain the same as shown below with an explanation in the comments.



**0 #Data Pipeline Architecture**

**#Creating a main diagram of the use case which consists of the further subcodes.**

**with Diagram(" Fumble - Your Personal Finance News Buddy ", show=True, filename=" FUMBLE") as diag2: #Creating the cluster containing the data sources and Mage**

**with Cluster( "Origin - Extract - Transform"):**

**#Creating a cluster of all the data sources which indicates a single source of data with Cluster("Data Sources"):**

**WSJ = Custorn("Wall Street Journal", "WSJ.png") MarketWatch = Custom( "MarketWatch", "marketwatch. png") CNBC = Custom("CNBC", "cnbc.png")**

**AlphaVantage = Custom( "AlphaVantage", "AlphaVantage. png") Mage = Custom("Mage", "Mage. png")**

#Creating the cluster containing the S3 and Elastic Search with Cluster("Load"):

db= S3("Store")

ElasticSearch = Custom("ElasticSearch","ElasticSearch.png")

#Creating the cluster containing the Destination resources and additional resources with Cluster("Destination - Transform"):

FastAPI = Custom("FastAPI", "Fast.png") Openai = Custom("Open AI", "OpenAI.png")

Streamlit = Custom("Streamlit", "Streamlit.png")

GoogleNews = Custom("Google News", "Google\_News.png")

#Designing the workflow by following the direction based on the use case. WSJ » Mage

MarketWatch >> Mage CNBC » Mage

AlphaVantage >> Mage Mage » db

db>> ElasticSearch

ElasticSearch >> FastAPI FastAPI >> ElasticSearch

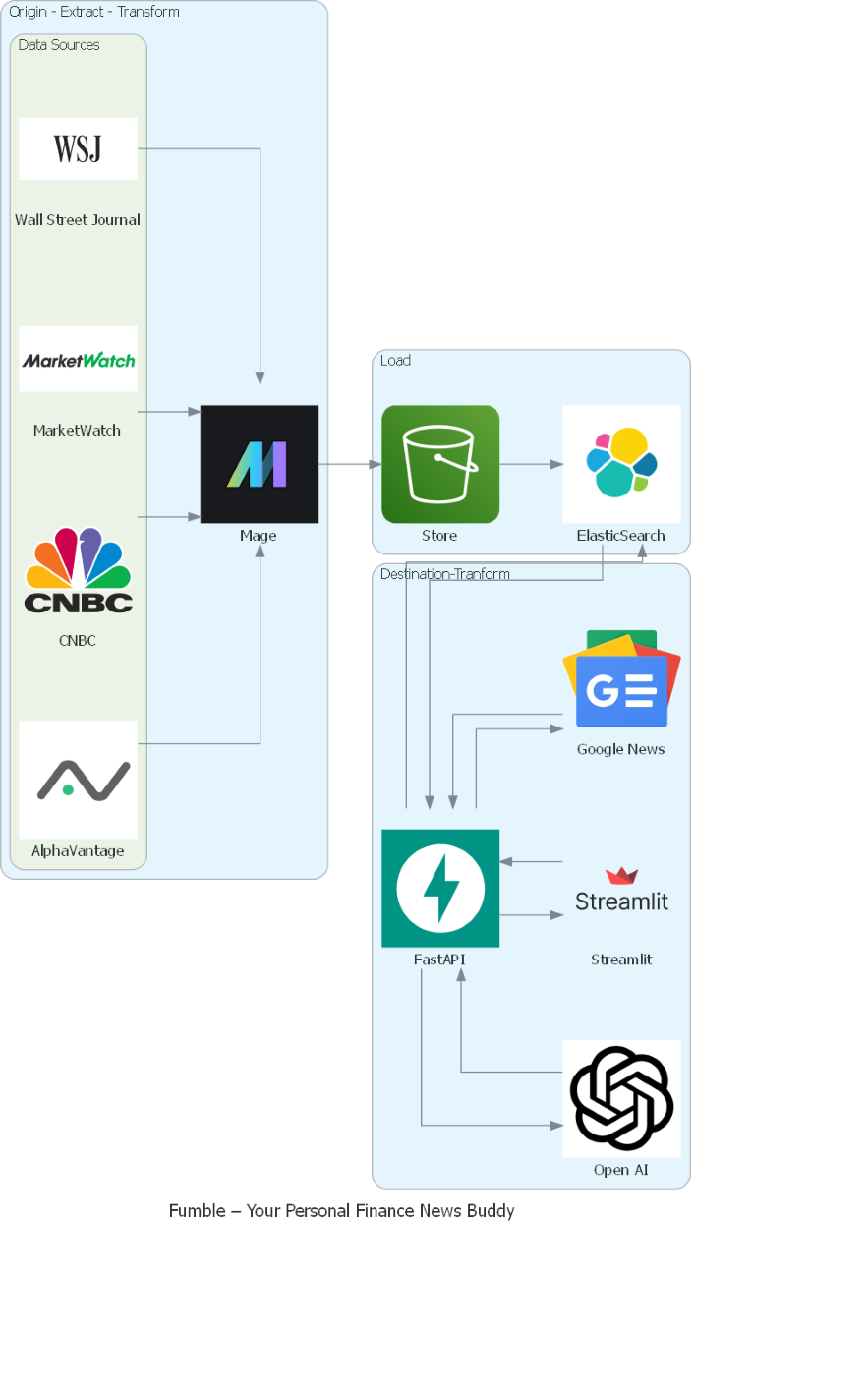
FastAPI >> Streamlit Streamlit >> FastAPI

FastAPI >> Openai Openai >> FastAPI

FastAPI >> GoogleNews GoogleNews >> FastAPI

#printing the finished diagram diag2

The architecture diagram with the utilized data pipeline pattern after executing the above code is as follows.



Resources which make up the ETLT Pipeline pattern.

* Origin – Data Sources
* Extract & Transform – Mage
* Load – S3 and ElasticSearch
* Transform – ElasticSearch, Open AI, Fast API
* Destination – Streamlit

The functionality of each component and justification these resources provide in the Fumble data pipeline:

* 1. *Data Sources*
     1. WSJ - Scrapes financial news articles from Wall Street Journal website
     2. MarketWatch - Extracts market and stock news data from MarketWatch
     3. CNBC - Pulls breaking finance headlines and market updates from CNBC
     4. AlphaVantage - Provides quantitative APIs for historical and real-time stock data
  2. *Mage*
     1. Centralized ETL engine that structures, cleans and processes all data from the various sources
     2. Handles joining disparate data sources, cleaning malformed content, feature extraction
     3. Outputs uniform dataset for downstream storage and consumption
     4. This acts as a first layer of transformation
  3. *S3 Data Store*
     1. Scalable object storage for holding substantial amounts of unstructured scraped data
     2. Serves as reliable data lake for pre-processed data before search indexing
     3. Decouples storage from compute for easier scaling
  4. *Elasticsearch*
     1. Indexes documents/articles to enable fast text search and analytics
     2. Optimized for text relevance ranking, filtering, aggregations
     3. Stores indexes for low-latency query serving by FastAPI
     4. Performs integration and other dependent works in the data storage, and this acts as a second layer of the transformation.
  5. *FastAPI*
     1. Provides REST API endpoints for client applications
     2. Serverless functionality enabled by integration with Elasticsearch index
     3. Handles business logic - search, summaries, recommendations, analytics
  6. *Streamlit*
     1. Creates intuitive web interface for end users
     2. Issues requests to FastAPI to fetch news feeds, summaries, and analysis
     3. Requires no JavaScript - quick iteration of data science models

### Specific Needs and Characteristics of Fumble

The data pipeline for Fumble is tailored to meet the needs of a personal finance news application with few specific characteristics and needs.

* Personalized News Delivery
* The integration of news sources, financial data, and summarization models allows for a customized news feed experience
* Elasticsearch enables rapid user profiling based on search and consumption patterns
* Streamlit interface can serve up articles matched to user interests
* Actionable Financial Insights
* Real-time and historical stock data from AlphaVantage which provides market analysis
* Summarization of market news events via OpenAI models
* Users can connect news to market performance and make investment decisions
* Intuitive User Experience
* Streamlit and OpenAI facilitate natural language interfaces without code
* FastAPI backend removes the data engineering complexity
* Users get simple, easy-to-understand financial summaries on demand

### Design Process and Decisions

The design methodology followed for building this pipeline is

1. Identify core use requirements and challenges that follow
2. Map required data inputs and outputs
3. Diagram high-level pipeline architecture
4. Detailed implementation choices for each component after comparing various resources which serve the same purpose.

Based on the use case, our pipeline requires,

* + Input – News articles, market data, user profile
  + Output – Summaries, search results, analytics High level Architecture

The initial architecture had the following few components and the resources which serve the purpose before dwelling deep as discussed above,

* + Data Source - (Wall Street Journal, Alpha Vantage, CNBC, Marketwatch)
  + Data Ingestion – (Spark, Airflow, Kafka, Mage)
  + Storage Layer – (HDFS, PostgreSQL, DynamoDB, AWS S3)
  + Text Processing Engine – (Solr, Lucene, Algolia, Elasticsearch)
  + API Serving Layer & Additional resources – (Fast API, Open AI, Google News)
  + Client interface - Streamlit

Data Source -> Data Ingestion -> Data Storage -> Text Search -> API Integration -> ML & Content generation -> Frontend

This high-level architecture led to the selection of the final resources after thorough research about the various resources which perform the same function.

(Wall Street Journal, Alpha Vantage, CNBC, Marketwatch) -> Mage-> AWS S3 -> Elastic-> Fast API (Open AI, Google News, Bing News) <-> Streamlit

Reasons for Eliminating other resources

* + ETL
    - Spark - More complex but scales better for big data
    - Airflow - More features for workflow management
    - Kafka Streams - Specialized for stream processing
  + Storage
    - HDFS - Distributed storage optimized for big data
    - PostgreSQL - Relational database, more structured
    - DynamoDB - Fully managed NoSQL database
  + Search
    - Solr - Popular open alternative to Elasticsearch
    - Lucene - Lower-level Java library underlying ES
    - algolia - Hosted search-as-a-service

Why we chose Mage + S3 + Elasticsearch

* + Simplicity & ease of use: Mage, S3, ES have simple APIs and integrate nicely
  + Cost: Pay-as-you-go pricing of S3 and ES cloud services
  + Community: Tested with lots of documentation available
  + Scalability: Ability to scale up these services as data volumes grow

# Assumptions Made

* + News sites and financial data providers have APIs or accessible webpages for scraping
  + The sources provide data in common formats like JSON, CSV, HTML
  + Daily data ingestion volume from sources will be <1 TB
  + Mage can handle the incoming ingestion for the expected volume
  + Pre-trained summarization models can effectively summarize short finance articles
  + Application will be hosted on cloud infrastructure (AWS)

**Use Cases to be supported by Fumble**

* + Generate daily market news digest emails customized to user preferences
  + Enable ad-hoc search to find articles on specific stocks/funds
  + Analyze news sentiment and correlate to portfolio performance
  + Serve a personalized finance article feed based on reading history

The above designed pipeline can accommodate all these use cases as,

* + Mage pipeline aggregates and filters the latest articles
  + Elasticsearch provides efficient full text search according to user preferences and history
  + Alphavantage keeps the updated portfolio data
  + FastAPI serves ranked content feed customized to user history
  + Streamlit displays paginated recommendations feed

**Open Questions**

* Can Mage keep up with future throughput as it is not well established in the market?
* How to evaluate if the selected resources are the most optimal for the pipeline?
* What kind of backup to be kept in place if a resource goes down?
* Personalization requires transparency and privacy, is it possible with AI models without human intervention?

# Future Improvements

* Check which resources get affected by substantial numbers of users and automate the setup of required resources to ease the maintenance.
* Add resources to remember common requests from the users.
* Build an easy way to restore data if needed and make the service reliable.
* Check the data quality to improve performance

# References

<https://blog.hubspot.com/marketing/press-release-template-ht> <https://mailchimp.com/resources/writing-press-releases/>

<https://www.forbes.com/sites/robertwynne/2016/06/13/how-to-write-a-press-release/?sh=1054bec23b93> <https://www.mage.ai/>

<https://diagrams.mingrammer.com/docs/getting-started/examples>

[https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html#:~:text=Elasticsearch%20provides](https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html#%3A~%3Atext%3DElasticsearch%20provides%20near%20real%2Dtime%2Cway%20that%20supports%20fast%20searches)

[%20near%20real%2Dtime,way%20that%20supports%20fast%20searches](https://www.elastic.co/guide/en/elasticsearch/reference/current/elasticsearch-intro.html#%3A~%3Atext%3DElasticsearch%20provides%20near%20real%2Dtime%2Cway%20that%20supports%20fast%20searches).