# Mandiri Securitas – Technical Test

Haikal Ramadhan Usman

# Task 1

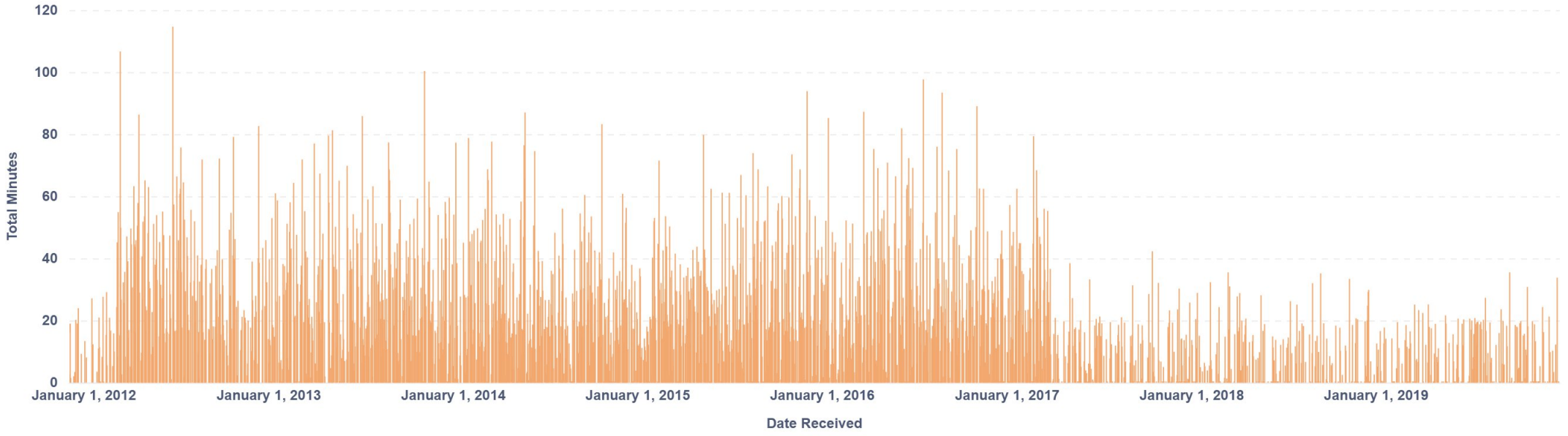
Data ingestion to Postgresql and generate dashboard summarize the average time to resolve complaints across a number of different dimensions.

# Average Call Duration

Before calculating the average call duration, data preprocessing was performed to remove all entries with a null *complaint_id*. The analysis also revealed that each *complaint_id* was unique. The final calculated average call duration was **11 minutes and 38.65 seconds** with the longest call duration was **28 minutes and 59 seconds.**

# Daily Total Duration

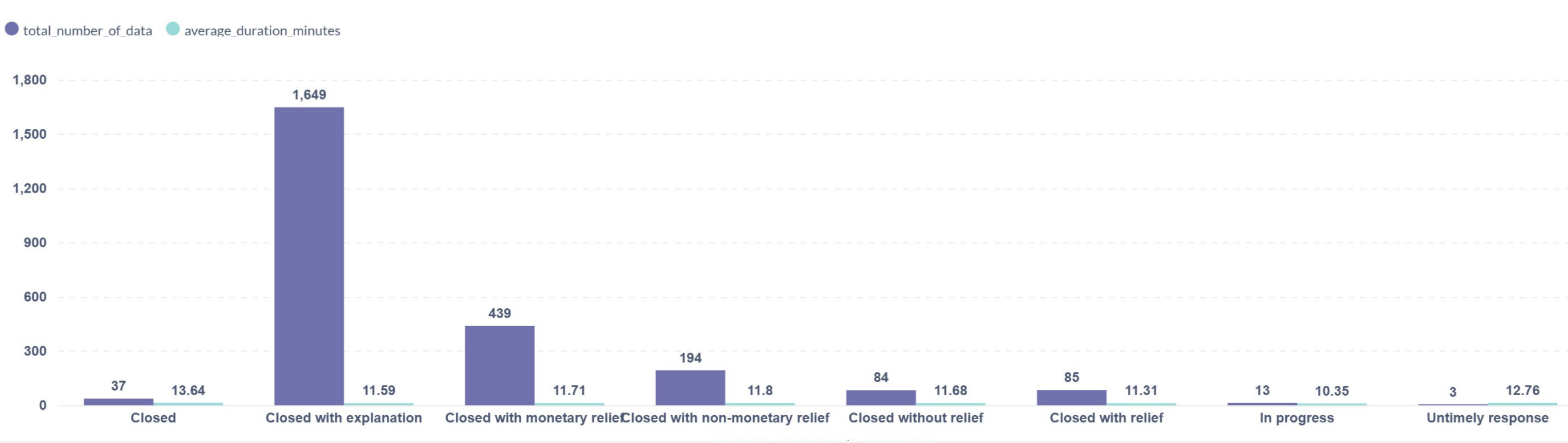
The chart shows total daily call duration in minutes from December 2011 to December 2019. Between 2012 and 2017, daily totals were relatively high and volatile, with peaks at 114.62 on June 25, 2012, indicating intense and inconsistent activity levels. Starting in early 2018, there was a sharp decline, with most daily totals rarely surpassing 40 minutes, suggesting a significant drop in the intensity or frequency of recorded activities.



# Total and Average call duration per Company Response

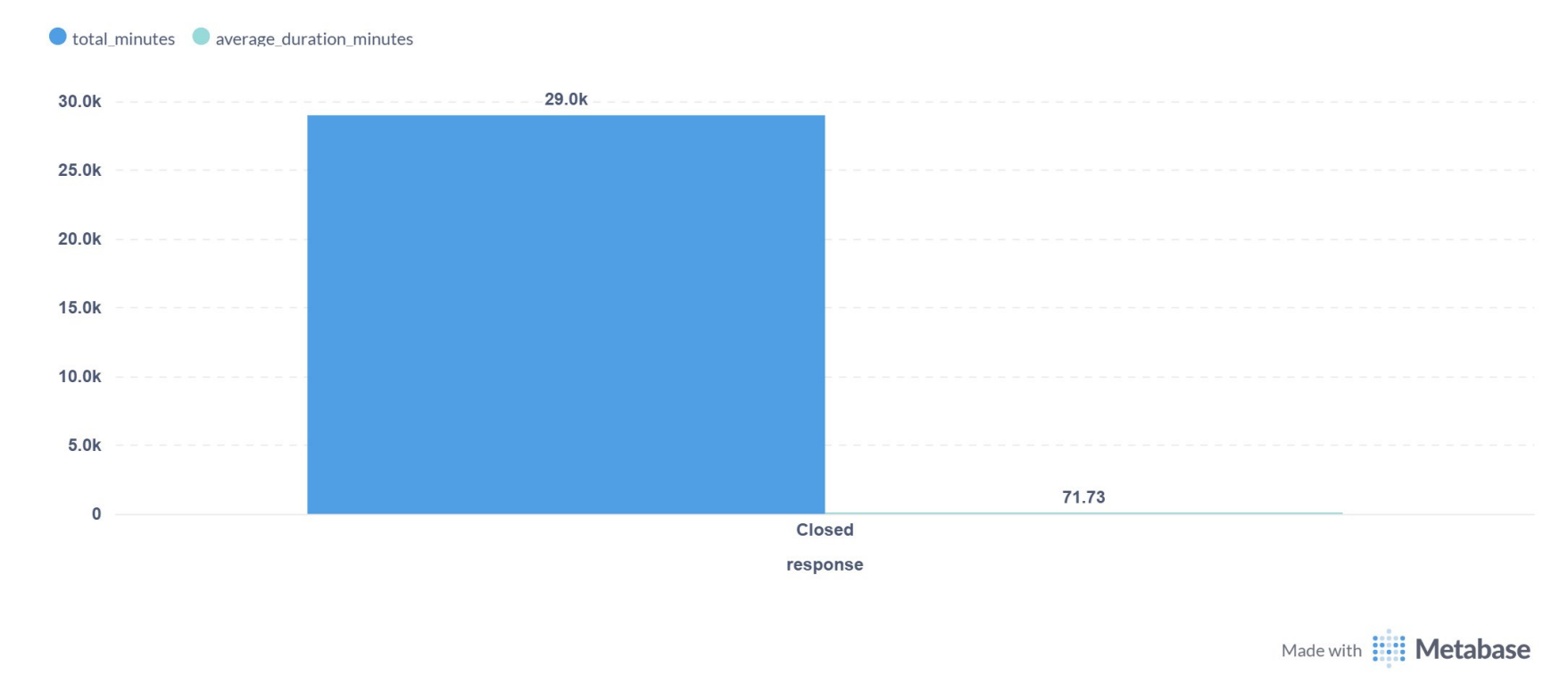
To determine the average time taken to resolve complaints, we first need to identify the different categories in **company_response_to_consumer**. From the chart below, eight categories were found.

- The fewest call duration was for *Untimely response*, with **total 3 cases** and an **average of 12.76 minutes call duration**. This shows that this case is very rare but need more time to solved
- The most call was *Closed with explanation*, with **1649 total cases** and an **average of 11.59 minutes call duration**. This suggests that a high volume of complaints could be resolved more quickly and efficiently if detailed explanations are provided.



# Average Time to Resolve Complaints

We have identified eight categories in *company_response_to_consumer*. We can now calculate the total and average duration to resolve complaints by summing all categories with the response “Closed.” The results show a total duration of **29,000 minutes** and an average of **71.73 minutes**.

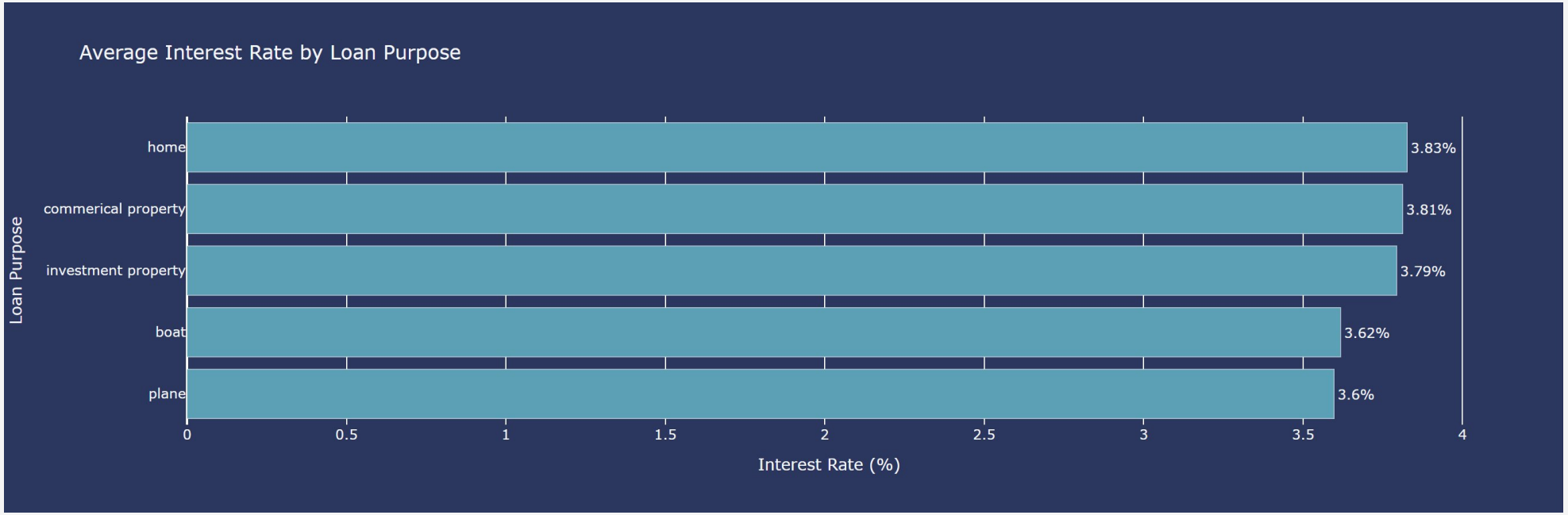


# Task 1

Data ingestion to Postgresql and generate dashboard summarize the average time to resolve complaints across a number of different dimensions.

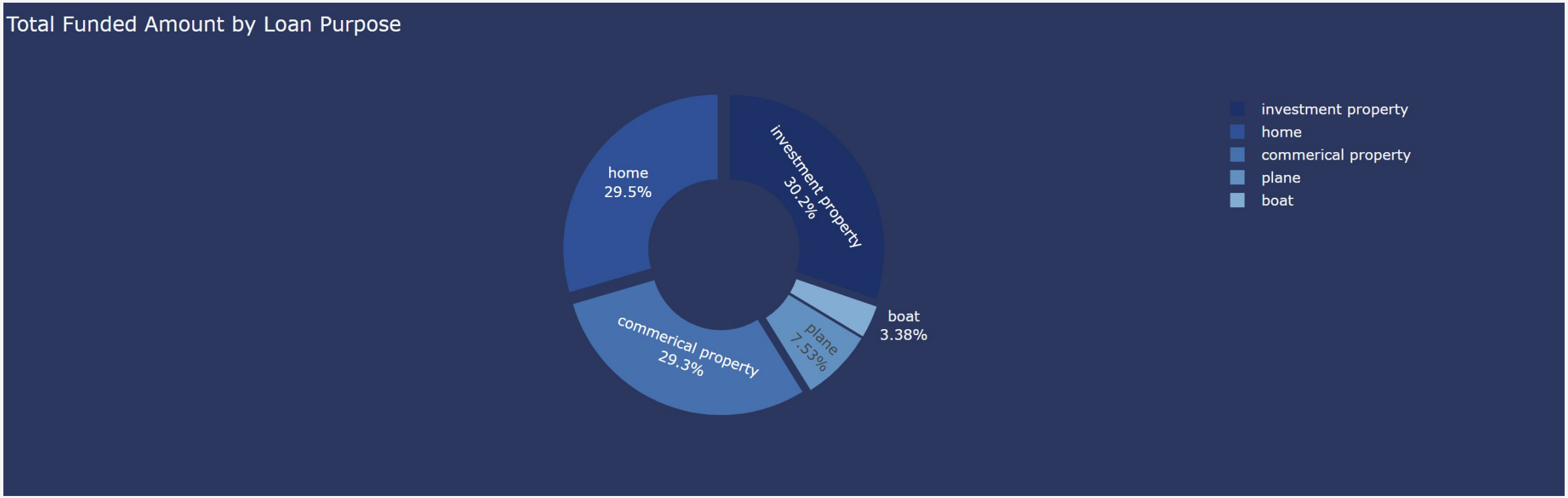
# Average Interest Rate by Loan Purpose

Shows the average of interest rate by the Loan purpose. Home is the highest interest rate with 3.83% and Plane is the lowest with 3.6%



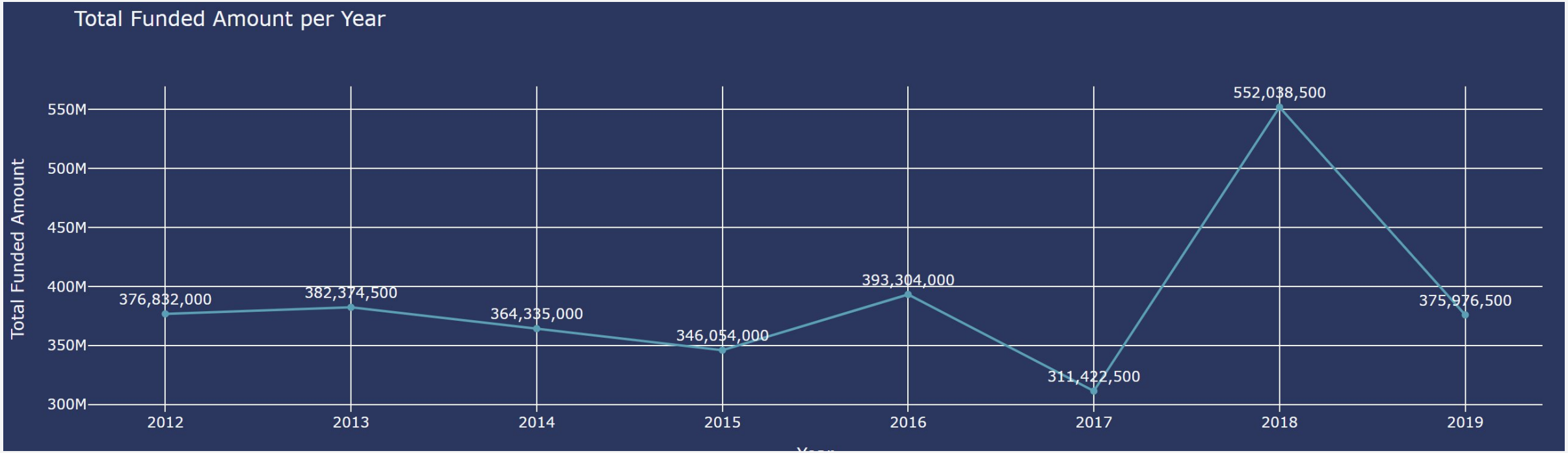
# Total Funded Amount by Loan Purpose

Show the Total Funded Amount by the Loan Purpose in percentage. If user hover the cursor to the pie chart, it will show the exact number. Investment property is the highest with 30.2% of total or 937912500. Boat is the lowest with 3.38% of total or 104963500.



# Total Funded Amount per Year

Showing the time series diagram of total funded amount per year. 2017 is the lowest with 311,422,500 total funded amount and 2018 become the highest with 552,038,500 total funded amount



# Task 3

How to generate LLM chat (chatgpt like solution or RAG (Retrieval-Augmented Generation)

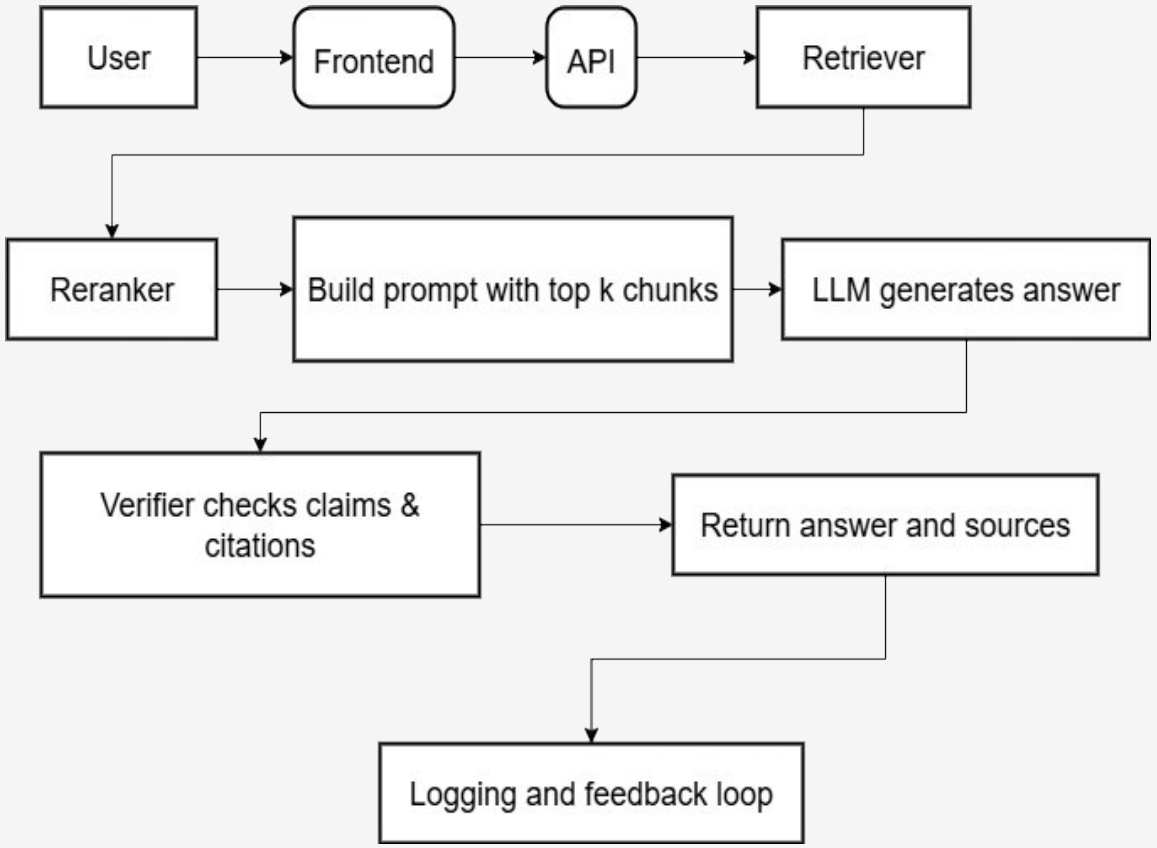
# Concept

Goal: Build an LLM-based chat that **only answers using internal company data** (DB tables, company documents), with responses that are:

- Accurate and traceable (citations to sources)
- Secure (no sensitive data leakage)
- Low hallucination rate
- Operable & maintainable (monitoring, feedback loop)

# Flow Diagram

1. **User -> Frontend**: Website or dashboard where user create prompt
2. **API**: Frontend send data to API
3. **Retriever**: Find relevant pieces of information
4. **Reranker**: Reorder information by the most relevant information.
5. **Build Prompt with top_k chunks**: Retrieve the top_k chunks
6. **LLM Generates Answer**: Generate Natural language answer
7. **Verifier Checks Claims & Citations**: Reduce hallucinations
8. **Return Answer + Sources**: API sends the final verified answer back to frontend
9. **Logging & Feedback Loop**:



# Detail Steps

1. **Data Preparation**
   - Identified Data source (DB, Documents, spreadsheet, API)
   - Designing data confidentiality level (Owner of the data and what action they can do for each document)
   - Provide read-only credential for systems
   - Generate ETL process to extract and cleaning data
2. **Chunking**: Split Document to fit within token limit
3. **Embedding & Vector Indexing**: Convert text to vector then organizing and storing the data. This will allow the system for fast and efficient searching
4. **Retrieve and reranking**: Getting the data and reorder it by the most relevance answer
5. **Prompt Engineering & RAG Orchestration**
   1. Build prompt by combining:
      1. The user Query
      2. top_k retrieved chunks
6. **LLM Generation**: Generate Natural language answer
7. **Evaluation**:
   - Run automatic script or use query to validate the answer
   - Create ambiguous question to test robustness
   - Create malicious prompts to check for data leakage
8. **Deploy**

# Avoid Hallucinations

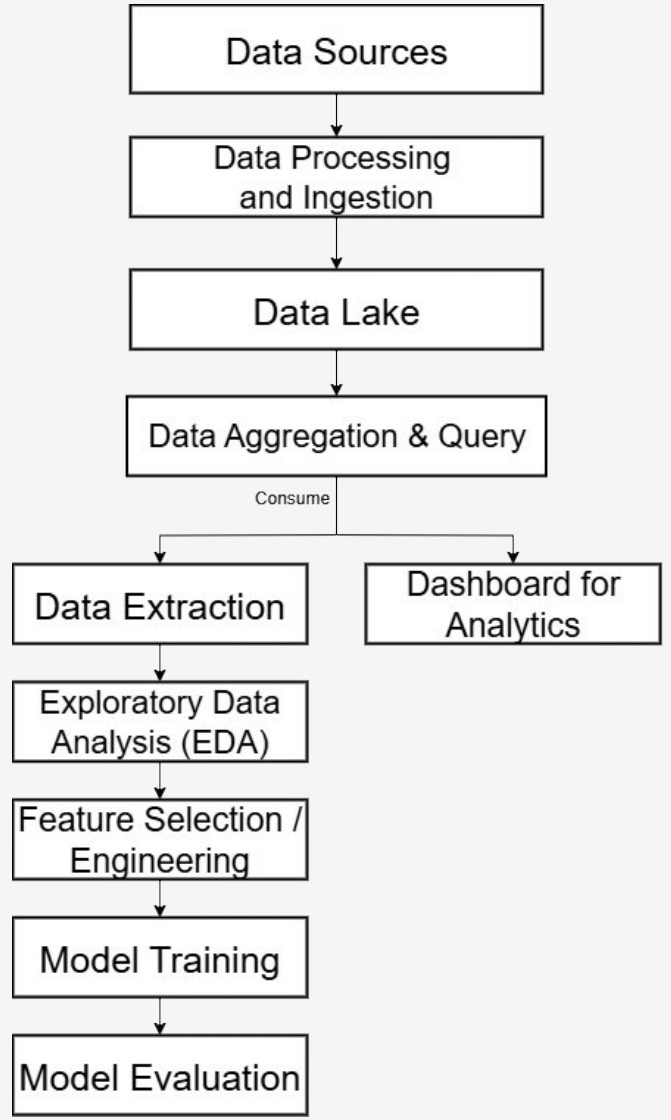
1. LLM should only answer based on chunks that retrieved from database. If not, LLM should return message “Information not available”
2. Use Lexical and Dense retriever combination to retrieve most relevant answer
3. Human feedback: asking feedback from user to improve the LLM
4. Add data or document source to validate answer.

# Task 4

Generate Data platform end-to-end system. Retrieve internal or external data and store it to cloud platform.

# Flow Chart

This Flow aims to build an end-to-end data platform capable of efficiently managing diverse internal and external data sources (Database, API, web scraping, spreadsheet, GA, Firestore, etc) starting from data ingestion using Airflow and processing with pyspark and storing raw data into Data Lake cloud storage such as AWS S3. The processed data serves two main purposes: first, as a foundation for business analysis and visualization through dashboard tools like Metabase, and second, as input for machine learning model development involving data exploration, feature selection, training, and evaluation to support predictive insights and automation in business decision-making. This approach ensures the platform delivers accurate, real-time insights while maintaining flexibility to accommodate both analytics and AI-driven needs, ultimately enhancing overall business value.



# Detail Step by Step

1. **Data sources collection**: Data sources could be internal or external such as, Databases, Backend API, Web scraping, Spreadsheet, PDF document, etc.
2. **Data Processing and Ingestion**: Process to orchestrate and schedule data ingestion to data lake using Apache Airflow. This will collect data from data sources. This process is also include Data cleaning and transformation with pyspark.
3. **Data Lake/Storage**: Store data from Data Ingestion process into cloud storage such as AWS S3. The files could be csv, json, etc
4. **Data Aggregation & Querying**: Use Aws Athena to create table based on data lake file and query directly to data lake
5. **Data Consumption**
    1. **Build ML Model**
        - **Data Extraction for Analysis and Modeling**: Extract aggregated data using Python tools.
        - **Exploratory Data Analysis**: Process of understanding data, including finding data pattern and understand quality of data
        - **Feature selection**: Selecting relevant feature for machine learning model
        - **Model Development**: Training ML model using prepared data. Use platform like Scikit-learn, TensorFlow, PyTorch, or cloud ML platforms like SageMake. This include performing hyperparameter tuning for optimization.
        - **Model Evaluation**: Evaluate model performance using appropriate metrics (accuracy,etc). Use cross-validation or test datasets for validation. Ensure the model is not overfitting and generalizes well.
    2. **Dashboard and Reporting**: Use BI platform such as Metabase to connect to AWS Athena and visualize insights and business metrics in real-time

# Task 5

Thoughts about the future of Financial Technology when it comes to investment banking

# Future of Financial Technology

In the future, Financial Technology will accelerate, with the help of AI and Machine Learning to transform how people analyze investment instruments and manage risk. This will make technical analysis less critical  highlight the importance of Data Science. However, the growth of fintech will increase the risk of cyber attacks. So, it would be wise for all actors in fintech industry to also prioritize the cybersecurity aspct.