DE Project Report

Group 12

Vivek Sapkal Kapil Yadav Bhagwan Arsewad

Advertisement Response Analysis

Github Repository Link: Advertisement-Response-Analysis

In this project, we designed a comprehensive system to analyze ad performance and user preferences using machine learning models. The project involved data collection, transformation, machine learning integration, and data visualization to provide actionable insights for targeted advertising. Screenshots of frontend pages are attached in the readme file of the github repository. Below is a detailed explanation of the technologies, data sources, and processes used.

Technology Stack and Purpose Justification

Frontend: ReactJS

Justification

React is chosen for its ability to handle dynamic and real-time data updates efficiently, which is crucial for displaying advertisement performance metrics. The **component-based architecture** allows the creation of reusable components for visualizations such as charts and graphs, ensuring consistency across the user interface. The **efficient Virtual DOM** minimizes unnecessary re-rendering, ensuring the application performs well even with large datasets and frequent updates.

Project Relevance

Given the need for continuous data visualization of various metrics (e.g., click-through rates, engagement time, and demographics), React enables fast and responsive updates to the interface, making it ideal for this project where user interaction with the charts is essential.

Backend: Django

Justification

Django was selected as the backend framework due to its **seamless integration with Python-based machine learning libraries**, such as **Scikit-Learn** and TensorFlow. This integration simplifies the process of serving machine learning models and running predictive analysis directly on the backend. Django's **built-in security features**, such as protection against SQL injection and cross-site scripting (XSS), ensure secure handling of sensitive user data, including demographic and purchase information. Additionally, **Django REST framework** allows us to expose the machine learning models and data processing pipelines through RESTful APIs, enabling smooth communication with the React frontend.

Project Relevance

The need for robust backend capabilities to manage user authentication, handle requests, and serve ML models made Django the ideal choice. It ensures the system can handle large volumes of advertisement data while maintaining security and performance.

Database: MongoDB

Justification

MongoDB was chosen for its **NoSQL structure**, which is well-suited to store semi-structured and diverse datasets, such as ad performance data, user responses, and demographic information. Since these datasets may vary in format, MongoDB's **document-oriented storage** allows flexible schema management, making it easier to scale and adjust the data structure as needed. Additionally, MongoDB's **horizontal scalability** ensures that as the project grows and more data is collected, the database can handle increasing volumes without performance degradation. MongoDB also supports **high-speed queries** and indexing, which is essential for quickly retrieving data for real-time analysis and reporting.

Project Relevance

Given the variety of data sources and attributes (e.g., ad platform types, demographics, user interactions), MongoDB's flexible schema and high-speed querying capabilities provide an ideal solution for storing and retrieving data for advertisement performance analysis.

Data Pipeline: ELT (Extract, Load, Transform)

Justification

The **ELT pipeline** approach was implemented to provide flexibility and efficiency in handling raw data. The pipeline allows us to **extract** data from various sources, such as surveys and online platforms, then **load** it into MongoDB, ensuring that no data is lost during the extraction process. After loading the raw data, **transformations** such as data cleaning, consistency checks, and redundancy management are applied, ensuring the data is of high quality and ready for analysis. This approach also supports real-time data transformations and cleaning, which is crucial for maintaining up-to-date and accurate datasets.

Project Relevance

The ability to handle and process raw data efficiently, applying necessary transformations as needed, enables the project to continuously ingest and process data without manual intervention. This is critical for maintaining data quality in dynamic advertising environments.

Containerization: Docker

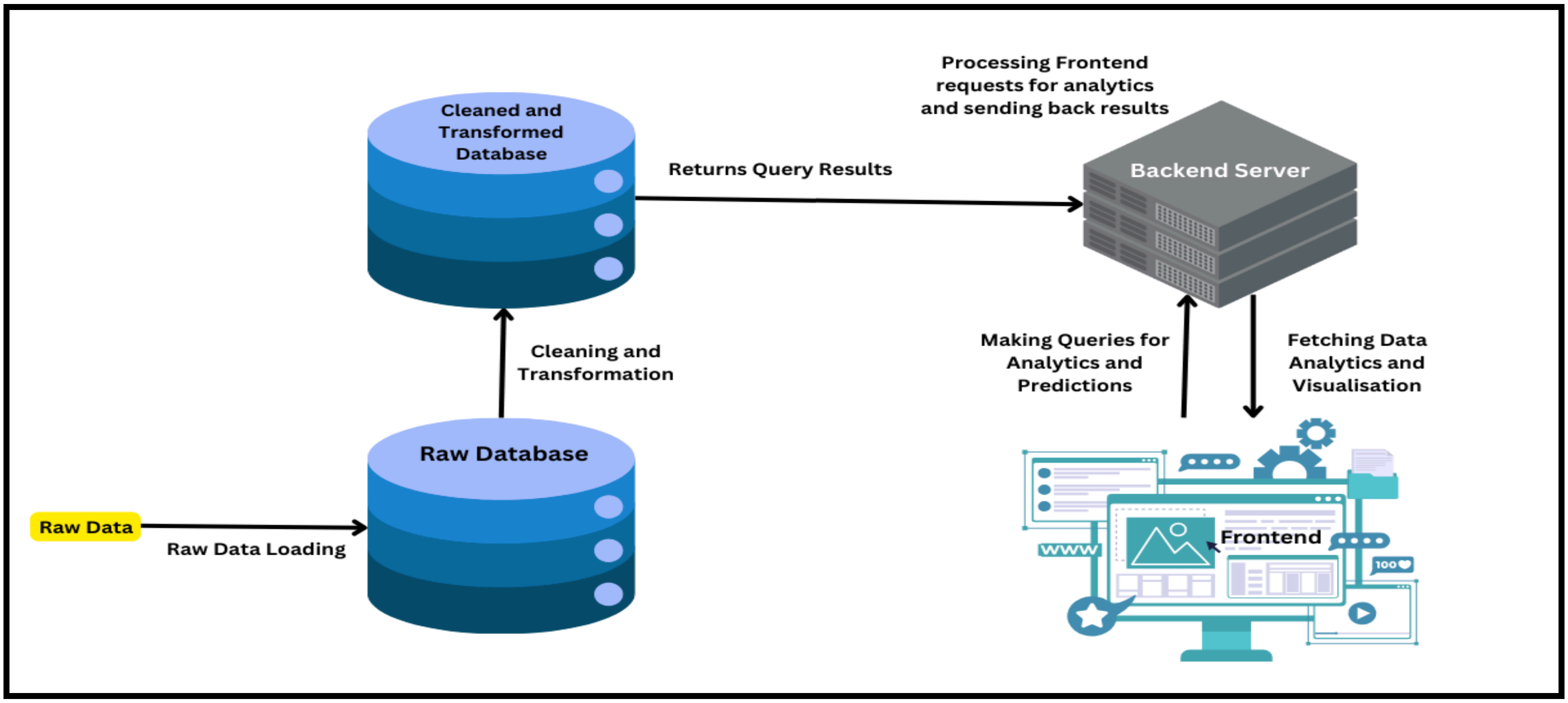
Justification

Docker was implemented to manage the application in a modular, containerized environment, enabling each part of the system (data loading, transformation, backend, frontend) to run independently. This approach allows for streamlined **scalability** and **efficiency**, as each container can be scaled or updated without affecting other parts of the system. Additionally, Docker ensures **consistent environments** across development, testing, and production, making deployment easier and more reliable. By separating concerns into different containers, Docker also simplifies **troubleshooting and debugging** of specific components.

Project Relevance

With the project involving multiple moving parts (data loading, cleaning, machine learning, and user-facing visualization), Docker's ability to modularize and encapsulate these processes ensures that each component can function independently and scale as needed, reducing deployment complexity and increasing efficiency.

Overall Project Architecture:


Processing Frontend requests for analytics and sending back results
Cleaned and Transformed Database
Returns Query Results
Backend Server
Cleaning and Transformation
Making Queries for Analytics and Predictions
Fetching Data Analytics and Visualisation
Raw Database
Raw Data
Raw Data Loading
Frontend

Data Collection: Raw data is fetched from online sources and Google Forms into the **Raw Data Database**.

Data Cleaning and Transformation: Scripts process the raw data, converting it into cleaned and structured formats, and store it in the **Cleaned Data Database**.

Visualization and Analysis:

- **Home Page**: Predefined charts are fetched and displayed using `chart_service.py`.
- **Analysis Page**: User-selected parameters trigger `dyn_chart_services.py` to generate and return customized charts.

Prediction: Users input attributes on the Predict Page, and `predict_services.py` uses ML models to generate predictions, returning results via APIs.

Containerized Deployment: Docker ensures isolated, reproducible environments for all components, streamlining updates and scalability.

Data Sources

1. Internet Data:

- We collected various datasets from websites focusing on advertisement metrics, user demographics, and engagement statistics. These sources provided the foundation for our ad performance prediction model, as they included information on metrics such as CTR, CR, ad costs, and user engagement with different ad types.
- **Data Collection Process:** Our team manually gathered data from trusted online sources, ensuring the relevance and authenticity of the information. This data was then structured into formats compatible with our database schema for seamless loading into MongoDB.

2. Peer Surveys:

- We designed a **Google form** to collect survey data from peers, capturing demographic details such as age, gender, and preferences. This peer data complements the web-sourced data by providing additional, localized demographic information. To view this data in google sheets, click here.
- **Synchronization with MongoDB:** The survey responses are stored in a google sheet linked with the google form, which is directly integrated with MongoDB using a python script and google sheets api. The data is regularly synchronized, with new entries added to the database after passing through the transformation and cleaning process.

Database Schema

Raw Database Schema

Table: `Survey_Respondents`

- **Purpose**: Capture demographic data of survey participants.
- **Attributes**:
  - `RespondentID`: Unique identifier for each respondent.
  - `Age`: Exact age of the respondent.
  - `Gender`: Male, Female, Other.
  - `Location`: Rural, Urban, Suburban, etc.
  - `IncomeLevel`: Monthly or annual income in specified ranges.
  - `EducationLevel`: High School, Undergraduate, Graduate, etc.
  - `Occupation`: Job title or type.

Table: `Advertisement_Info`

- **Purpose**: Store metadata about advertisements.
- **Attributes**:
    - `AdID`: Unique identifier for each ad.
    - `AdPlatformType`: Type of platform (e.g., TV, Social Media).
    - `AdPlatformName`: Specific platform (e.g., Google, Facebook).
    - `AdType`: Video, Banner, Text.
    - `AdTopic`: Topic or theme of the ad.
    - `AdDuration`: Length of the ad in seconds.
    - `AdCost`: Cost associated with the ad.
    - `PurchaseAmount`: Revenue generated from purchases due to the ad.

Table: `Responses_to_Ads`

- **Purpose**: Record user responses to advertisements.
- **Attributes**:
    - `RespondentID`: Links to `Survey_Respondents`.
    - `AdID`: Links to `Advertisement_Info`.
    - `ResponseDate`: Date the ad was seen.
    - `ResponseType`: Clicked, Ignored.
    - `PurchaseIntent`: Yes/No indicating intent to purchase.
    - `Relevant`: Indicates whether the respondent found the ad relevant.
    - `EngagementTime`: Time spent engaging with the ad (seconds).
    - `Rating`: User rating of the ad (scale 1-5).
    - `DeviceType`: Mobile, Desktop, Tablet.

Table: `Ad_Demographic_link`

- **Purpose**: Link advertisements to specific demographics.
- **Attributes**:
    - `AdID`: Links to `Advertisement_Info`.
    - `DemographicID`: Links to `Demographic_Data`.

Table: `Demographic_Data`

- **Purpose**: Standardized demographic groups.
- **Attributes**:
    - `DemographicID`: Unique identifier.
    - `AgeGroup`: E.g., 18-25, 26-35.

    - `Gender`: Male, Female, Other.
    - `Location`: Rural, Urban, Suburban.
    - `IncomeLevel`: Specified ranges.
    - `EducationLevel`: High School, Undergraduate, etc.

Table: `Purchase_Info`

- **Purpose**: Store purchase behavior data.
- **Attributes**:
    - `RespondentID`: Links to `Survey_Respondents`.
    - `AdID`: Links to `Advertisement_Info`.
    - `InfluenceFactor`: Reason for purchase (e.g., Peer Recommendation).
    - `PurchaseLocation`: Online or In-Store.
    - `AdInfluence`: Yes/No indicating whether the ad influenced the purchase.

Transformed Database Schema Changes:

Age Group, Income Level and **Engagement Time:** are converted to categorized ranges from integer values.

CTR and Conversion Rates: Derived metrics. A new mongodb collection is added to store ctr and conversion rate of every ad. This table is updated time to time by the cleaning and transformation script.

Ad Duration: is set to zero non-video type ads.

Adcost and Purchase Amount: is changed to integer from float values.

Data Loading and Transformation Process

To maintain data quality and readiness for analysis, we employed an **ELT (Extract, Load, Transform) pipeline**. Here's a breakdown of each step:

Data Extraction and Loading

- **Extraction:** Data is pulled from websites and the Google Form responses.
- **Loading:** Data is initially stored in MongoDB in its raw format to preserve the original dataset. MongoDB's flexibility allows us to store varied data types without

extensive preprocessing, making it easy to handle new data from different sources.

Data Cleaning and Transformation

After loading, data undergoes a transformation process to ensure its quality and consistency. The steps involved include:

- **Data Consistency:** We removed duplicate entries, standardized data formats (e.g., date and currency formats), and resolved any discrepancies between different datasets to ensure accuracy.
- **Data Completeness:** Missing values were handled through imputation techniques or were flagged if unresolvable. For example, if demographic information was missing, we either estimated it based on related entries or flagged the entry as incomplete.
- **Data Redundancy Management:** Redundant data was identified and removed to improve data efficiency and storage optimization. This step helps avoid repetitive information, ensuring that only necessary data is retained for analysis.

The data is fetched from the database transformed and cleaned according to above steps and then fed to another mongodb database which only stores cleaned and transformed data.

Data Analysis Results and ML Models

Our project leverages machine learning models to perform in-depth analysis and provide predictive insights on advertisement performance. We designed and implemented two machine learning models: one focusing on ad performance metrics and another on user demographic preferences.

Ad Performance Prediction Model

This model is a **regression model**. This model is built to predict two primary ad performance metrics:

Click-Through Rate (CTR): Measures how often people click on an ad after seeing it.

Conversion Rate (CR): Measures the percentage of users who take the desired action (like making a purchase) after clicking the ad.

Input Attributes for Prediction:

- **Ad Cost:** The budget allocated to the ad campaign, which can influence CTR and CR.

- **Ad Topic:** The category of product (e.g., Food, Medicine) as certain product types may have different engagement levels.
- **Ad Type:** Whether the ad is a video, banner, or other type, which influences engagement rates.
- **Broadcast Platform:** The medium where the ad will appear (e.g., TV, social media, or website), impacting how viewers interact with it.
- **Broadcast Platform Name:** The name of the platform where the ad will appear (e.g. google, netflix, etc).
- **Purchase Amount:** The purchase amount of the product being advertised.

Model Objective: The goal of this model is to estimate the potential success of an ad campaign. By providing predicted CTR and CR, advertisers can gauge the ad's effectiveness and make informed decisions about whether to proceed with a campaign. This predictive insight allows businesses to strategically allocate their ad budgets based on expected engagement.

Example Use Case: For a given ad campaign with specified attributes (e.g., product type as "Food," ad type as "Video," and broadcast platform as "Social Media"), the model would output an expected CTR and CR. Businesses can then compare these metrics across multiple platforms or ad types to choose the optimal combination for maximizing engagement.

User Demographic Ad Preference Model

This model uses demographic data to predict which type of ad a user is more likely to respond to using a **decision tree model**. By understanding ad preferences based on user demographics, businesses can target ads to specific audiences more effectively, thereby enhancing ad relevance and potential impact.

Input Demographic Attributes:

- **Age:** Different age groups may respond differently to ad types (e.g., younger people may engage more with video ads).
- **Gender:** Gender-based preferences can impact which ads are more appealing.
- **Location:** Rural, Suburban and Urban locations can have different preferences for ads.
- **Occupation and Income Level:** These factors indicate a user's purchasing power and influence preferences.
- **Education Level:** Higher or lower education levels may correlate with different types of ad engagement.

Model Objective: The purpose of this model is to allow advertisers to identify the most appropriate demographic for a given ad type. By predicting user preferences, businesses can tailor ad campaigns to the most receptive audience, optimizing ad reach and reducing ad spend waste.

Example Use Case: For a target demographic of users aged 25-34 with high education and mid-level income, the model might suggest that video ads for tech products are most likely to resonate with this group. This information helps advertisers position their ads for maximum impact, thereby increasing the likelihood of conversions.

Bias and Fairness Analysis

Bias and fairness were important considerations in designing our ML models. To ensure unbiased and fair predictions, we implemented the following strategies:

Data Imbalance Correction

We examined the data for demographic representation and addressed any imbalances that could lead to biased model recommendations. For example, if one gender or age group was overrepresented, we balanced the training data to ensure fair predictions.

Evaluation Metrics (Precision and Recall)

Precision and recall were used to assess model performance across different demographics, helping us identify and rectify any significant disparities in the model's performance. For instance, if recall was notably lower for certain demographic groups, we adjusted the model or dataset to improve its sensitivity to those groups.

By implementing these strategies, we minimized biases that could result from demographic imbalances, ensuring that the models offer fair and equitable recommendations.

Data Visualization

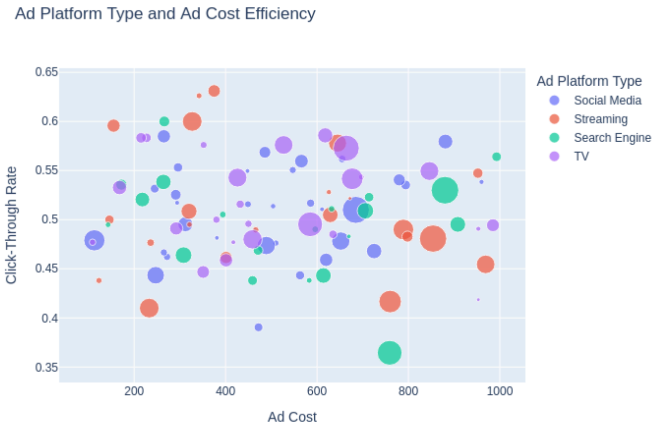
The data visualization component is integral to our project, providing users with tools to explore insights interactively and visualize ad performance metrics. We focused on creating charts and graphs that allow users to analyze the effectiveness of ads across different platforms, demographic segments, and ad types.

Key Visualizations

Ad Platform Type and Ad Cost Efficiency

Purpose

To visualize the relationship between ad cost, click-through rate (CTR), and engagement time across different ad platform types.



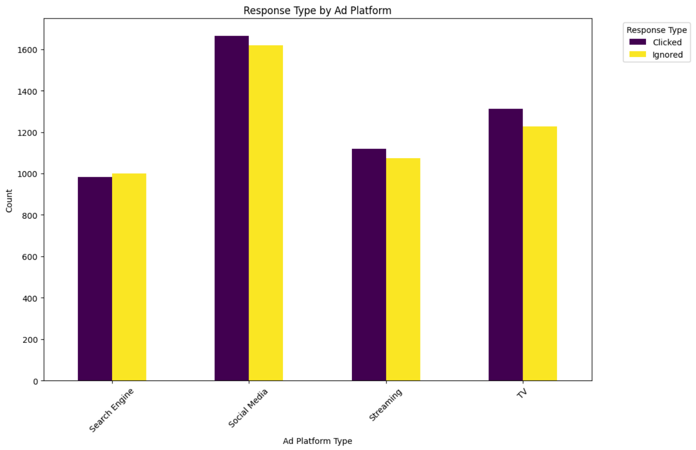
Description

This bubble chart uses AdCost on the x-axis, Click_Through_Rate (CTR) on the y-axis, and the size of the bubbles represents Mode_Engagement_Time. The color of the bubbles distinguishes between different AdPlatformType values (e.g., Social Media, TV, Streaming, and Search Engine). The chart helps to identify which ad platform provides better cost efficiency and engagement for the given ad spend.

Example Insight

By analyzing this chart, we can quickly identify which ad platform offers the best return on investment (ROI) by comparing ad cost to engagement time and click-through rates. For instance, if Social Media ads show a lower cost with a higher engagement time and click-through rate, it might indicate a cost-effective platform for advertisers.

Response Type by Ad Platform



Purpose

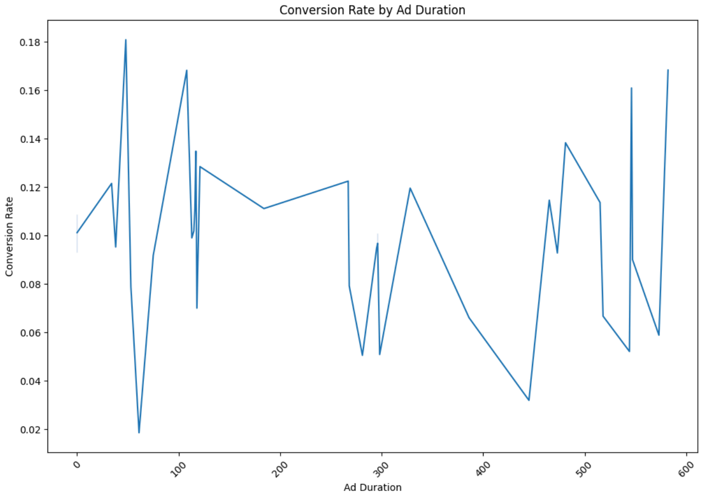
To compare the click-through rate (CTR) across different ad topics.

Description

This bar chart displays the average click-through rate (CTR) for various ad topics. It highlights the relative effectiveness of different ad themes in driving user engagement. Each bar represents the CTR for a particular AdTopic, allowing for an easy comparison across all topics.

Example Insight

This chart can help advertisers pinpoint which ad topics resonate with their target audience. For example, if ad topics like "Technology" or "Health" yield higher CTRs, advertisers can tailor their campaigns around these topics to increase engagement and improve overall performance.



Conversion Rate by Ad Duration

Purpose

To explore the trend in conversion rate as a function of ad duration.

Description

The line chart plots the conversion rate against ad duration. It provides a clear view of how ad length affects the likelihood of a user converting after viewing an ad. The x-axis represents the ad duration, while the y-axis shows the conversion rate. This visualization helps to understand the optimal duration for ads to maximize conversion.

Example Insight

Advertisers can use this chart to optimize their ad durations. For instance, if the chart shows that ads with a duration around 30 seconds have the highest conversion rates, it suggests that this length is optimal for their campaigns. This insight can be used to refine ad strategies, reducing ad spend while maximizing conversion potential.

Interactive Analysis

Dynamic Frontend Analysis Page

The **Analysis Page** in the React frontend allows users to select chart types, attributes, and filters dynamically. This flexibility lets users visualize data in various forms, such as bar, line, pie, or scatter plots. Filters for demographic, engagement, or advertisement characteristics enhance data exploration. Charts update in real-time via REST API calls, ensuring responsiveness and seamless user experience.

Backend

The dyn_chart_services module processes user-selected parameters, dynamically generates MongoDB queries, and aggregates results using pandas for efficient data transformation. It supports multiple chart types and provides additional summary metrics like CTR, engagement scores, and demographic trends. Optimizations like caching and indexing ensure high performance, even for complex queries.

Novel Idea Explored

Dynamic Data-Driven Chart Generation

One of the novel ideas explored in this project is the implementation of a dynamic data visualization system that automatically determines the most suitable chart type and analysis based on user-selected attributes from the database. Unlike static or pre-defined dashboards, this system leverages metadata and intelligent rules to create contextually relevant visualizations. This allows for greater flexibility and personalization, enabling stakeholders to derive insights tailored to their specific queries without manual intervention or coding. This provides unique insights into ad performance.

Comparisons with Existing Tech Solutions

1. Google Analytics: Focuses on web-based user behavior, whereas this project analyzes diverse ad platforms and integrates performance metrics like CTR and cost efficiency, utilizing MongoDB's flexible schema.

2. Tableau: Requires manual chart setup; this project automates dynamic visualization generation based on user-selected attributes, ensuring simplicity and accessibility for non-technical users.

3. HubSpot Ads Analytics: Offers CRM integration but lacks advanced metrics like cost efficiency and trends in conversion rates by ad duration, which this project incorporates.

4. Power BI: Provides custom visuals, but this project's automated chart creation and Dockerized environment enhance scalability and usability for data-driven decision-making.

Key Differentiation: Combines dynamic visualizations, cross-platform ad analysis, and cost efficiency metrics in a flexible Python-MongoDB-React ecosystem, surpassing predefined dashboards in adaptability.

Conclusion

This project successfully combines advanced data analysis, automation, and dynamic visualization to provide actionable insights into advertisement performance. By leveraging a modern technology stack—React, Django, MongoDB, and Docker—the system ensures seamless integration, scalability, and real-time data processing. The automated chart generation feature simplifies data interpretation, making it accessible even to non-technical stakeholders.

Through insightful metrics such as cost efficiency, click-through rates, and conversion trends, the project delivers a deeper understanding of advertising strategies' effectiveness across diverse platforms. Moreover, the innovative approach of integrating dynamic visualizations and real-time transformations distinguishes this solution from traditional ad analytics tools, emphasizing adaptability and decision-making support.

Overall, the project offers a robust framework for comprehensive advertisement analysis, paving the way for more informed marketing strategies and optimizing ad investments in a data-driven manner.

End of Report