Hochschule Fulda
University of Applied Sciences

Stream Processing and Analytics of Austin Bike Share Data

Shinu Donney | Usama Tahir

Technical Report
for the module *Stream Processing and Analytics*
in the study program *M.Sc. Global Software Development*
at Fulda University of Applied Sciences

Matriculation Numbers: 1457966,1453517
Supervisor: Prof. Dr. David James

Submitted September 2, 2023

Abstract

This technical report presents a comprehensive study on the stream processing of Austin Bike Share data, aimed at efficiently and effectively handling the real-time influx of information from the bike-sharing system. With the increasing popularity of bike-sharing programs, it has become essential to analyze and process the continuous flow of data generated by the numerous bike stations and riders.

The report begins by outlining the architecture and design of the stream processing system, which incorporates Google PubSub as the messaging system and Apache Beam for stream processing. We delve into the reasoning behind the choice of these technologies and their integration into the data pipeline.

Next, we discuss the data sources and data schema used in the stream processing application. The Austin Bike Share dataset includes information on bike stations, trip details, user information, and geospatial data, all of which require thoughtful handling and processing in real-time.

The core focus of this report revolves around the various stream processing techniques and operations applied to the data streams. We investigate key aspects such as data ingestion, filtering, aggregation, enrichment, and event-time processing. Additionally, we explore methods for dealing with late events and out-of-order data to ensure the accuracy and integrity of the processed results.

Furthermore, performance evaluation and optimization strategies are presented, showcasing how the stream processing system can efficiently handle the continuous data flow with low latency and high throughput. Scalability aspects and the ability to cope with varying data volumes are also addressed.

Finally, we present a range of use cases and practical applications of the stream processing system, including real-time analytics, anomaly detection, and dynamic visualization of bike share usage patterns. These applications demonstrate the benefits of processing the data in real-time, providing valuable insights for bike-sharing operators, city planners, and users alike.

In conclusion, the technical report provides a comprehensive understanding of the stream processing architecture and techniques used for Austin Bike Share data. It highlights the significance of real-time data analysis in bike-sharing systems and demonstrates how the proposed stream processing solution effectively handles the continuous data streams, enabling timely and informed decision-making for bike-sharing operations.

Contents

List of Figures **IV**

List of Tables **IV**

1 Introduction **1**
1.1 Context . . . 1
1.2 Problem statement . . . 2
1.3 Goals . . . 3

2 Requirment Analysis **4**
2.1 Functional Requirements . . . 4
2.2 Data Security and Compliance Requirements . . . 5
2.3 Deliverables . . . 5

3 Implementation **5**
3.1 Data Extraction and Preparation . . . 5
3.2 Real-time Streaming using Google Cloud Pub/Sub . . . 5
3.3 Streaming Analytics using Python . . . 6
3.4 Data Visualization . . . 6

4 Workflow **6**
4.1 Data Source Selection . . . 6
4.2 Python Script for Data Ingestion . . . 6
4.3 Publishing Data to Pubsub Topic . . . 6
4.4 Python Pipeline for Data Transformation . . . 7
4.5 Storing Processed Data on BigQuery . . . 7
4.6 Testing and Optimization . . . 7
4.7 Interactivity and Insights . . . 7

5 Experiments **8**
5.1 Type of Bikes Used in Trips . . . 8
5.2 Bikes With Most Trips . . . 8
5.3 Most Used Bikes . . . 9
5.4 Busy Stations . . . 11
5.5 Busy Hours . . . 12

6 Code **13**
6.1 Code for Publisher . . . 13
6.2 Code snippet of main pipeline . . . 15
6.3 A helper class which contains the logic to translate the user's custom dict into a BigQuery table row and then write it to BigQuery . . . 18

6.4 This code contains a pipeline that calculates the number of trips by subscriber type in a given time window 19
6.5 Github link to complete code base developed and maintained actively . . 20

7 Discussion **20**

8 Conclusion **21**

List of Figures

1 Workflow Diagram 8

List of Tables

1 Type of bike used in trips 8
2 Bike used for most trips in first week of April 2023 9
3 Bike used for most time in first week of April 2023 10
4 Most busy stations in 1st week of April 2023 11
5 Hourly data for 15/April/2023 12

1 Introduction

In the vibrant city of Austin, Texas, where innovation meets a commitment to sustainable urban mobility, the bike share program has emerged as a cornerstone of modern transportation solutions. This program facilitates the convenience of biking for both residents and visitors, promoting a greener and more accessible way to navigate the city's bustling streets. At the heart of this dynamic initiative lies a wealth of data that encapsulates the essence of how people move, interact, and engage with this two-wheeled mode of travel.

The Austin bike share data, made readily accessible through the Google Cloud Platform (GCP) public dataset, is a treasure trove of insights waiting to be unearthed. This dataset offers a comprehensive view of bike share activities, encapsulating details such as trip origins and destinations, user profiles, trip durations, and broader usage trends. These data points represent the pulse of urban mobility, a digital representation of the pathways that connect people, places, and experiences.

We seek to discern the patterns that emerge from the ebbs and flows of daily life, the factors that influence when and where people choose to pedal, and the impact of external forces like weather and events on these decisions. Through rigorous analysis, we aim to uncover not only the who, what, and where of bike share utilization but also the "why" that underlies these behaviors.

Our exploration is empowered by the Google Cloud Platform's robust suite of tools and technologies. The BigQuery data warehouse grants us the ability to query and manipulate large datasets with unprecedented speed and flexibility, while Google Data Studio allows us to transform raw numbers into visual narratives that can be easily shared and understood. These resources, combined with machine learning capabilities, position us to extrapolate insights beyond the surface and into the realm of prediction, enabling us to anticipate demand, optimize routes, and enhance user experiences.

1.1 Context

In Austin,Texas bike sharing has become an integral part of the urban transportation landscape, providing residents and visitors with an eco-friendly and convenient mode of travel. The city's bike share program has amassed a vast amount of data over time, capturing details about bike usage patterns, trip durations, user demographics, and more. To harness the insights hidden within this data, transportation analysts and data scientists have turned to the Google Cloud Platform's (GCP) public dataset resources.

The GCP public dataset for Austin bike share contains anonymized and aggregated data collected from the bike share program. This dataset encompasses various dimensions of information, including:

- **Trip Information**: Details about individual bike trips, such as start and end times, trip durations, start and end station IDs, and bike IDs.

- **User Demographics**: Generalized information about users, including age groups, gender distribution, and subscription types (casual users vs. registered members).
- **Geographical Data**: Information about bike stations, their locations, and associated metadata.
- **Usage Patterns**: Aggregated statistics on hourly, daily, and monthly usage patterns, helping to identify peak usage hours, popular routes, and seasonal trends.
- **Weather Data**: Supplementary weather information like temperature, humidity, and precipitation that can be correlated with bike usage patterns.

1.2 Problem statement

The objective is to extract meaningful insights, patterns, and trends from the collected data to facilitate data-driven decision-making, improve the efficiency of the bike-sharing system, and enhance user experience. The primary objectives are as follows:

- Data Collection and Preparation: Gather and preprocess the bike share data, including information on bike stations, trip details, user demographics, geospatial data, and any other relevant metadata. The data should be cleaned, organized, and made ready for analysis.
- Usage Pattern Analysis: Analyze the bike share data to identify usage patterns, peak hours, and popular routes. Determine the most frequently used bike stations and uncover factors contributing to usage fluctuations over time.
- User Behavior Study: Conduct an in-depth investigation of user behavior, including trip durations, user demographics, and preferences. Determine if certain user groups have distinct usage patterns or preferences.
- Seasonal and Spatial Variations: Explore how bike share usage varies across different seasons, weather conditions, and geographic regions. Identify regions with higher demand and assess the need for station distribution optimization.
- System Performance Evaluation: Evaluate the performance of the bike-sharing system in terms of bike availability, station utilization, and response to demand. Identify potential bottlenecks and areas for improvement.
- Key Performance Indicators (KPIs) Assessment: Define relevant KPIs for the bike-sharing program and assess its performance against these metrics. Compare the actual performance with predefined targets or industry benchmarks.
- Comparative Analysis: Conduct a comparative analysis with other bike-sharing programs or cities to benchmark the performance and effectiveness of the analyzed bike share program. Identify best practices and areas for potential improvement.

- User Satisfaction and Feedback: Investigate user feedback and satisfaction levels through surveys or data analysis. Determine factors influencing user satisfaction and identify areas for service enhancements.

- Data Visualization: Present the findings and insights in a visually appealing and easily interpretable manner through data visualizations, charts, and graphs. Provide intuitive visual representations of usage patterns and trends.

- Recommendations: Based on the analysis results, propose actionable recommendations to optimize the bike-sharing system's operations, improve resource allocation, enhance user experience, and address any identified shortcomings.

The report's ultimate goal is to offer a comprehensive analysis of the bike share data that can serve as a foundation for data-driven decision-making, infrastructure planning, and service improvements to ensure the success and sustainability of the bike-sharing program.

1.3 Goals

- Insight Generation: The primary goal is to gain valuable insights into the bike-sharing system's usage patterns, user behavior, and overall performance. The report aims to uncover hidden trends and correlations in the data that can be used to make informed decisions.

- Optimization Opportunities: Identify opportunities for optimizing the bike-sharing system's operations, such as bike station placement, bike redistribution strategies, and resource allocation, to enhance overall efficiency and user satisfaction.[1]

- Demand Forecasting: Develop predictive models to forecast future bike demand at different stations and times, aiding in proactive management of bike availability and ensuring an optimal user experience.

- User Profiling: Create user profiles based on demographic data and usage patterns, helping in tailoring marketing efforts and user-specific promotions to increase ridership and loyalty.

- Performance Evaluation: Assess the bike-sharing program's performance against predefined key performance indicators (KPIs) to determine its success in meeting set targets and industry benchmarks.

- Anomaly Detection: Implement anomaly detection techniques to identify irregularities in bike usage, station operations, or system performance, enabling prompt actions to address issues and maintain service reliability.

- Seasonal and Geographic Analysis: Analyze how bike usage varies across different seasons and geographic regions, allowing for targeted interventions in areas with high demand or underutilized stations.
- Decision Support: Provide data-driven recommendations and actionable insights to stakeholders, city planners, and bike-sharing operators, supporting them in making well-informed decisions regarding system enhancements and expansion.
- Comparative Analysis: Conduct a comparative analysis with other bike-sharing programs or cities to benchmark performance, identify best practices, and adopt successful strategies from other successful implementations.
- Policy and Infrastructure Planning: Inform policy-makers and city officials about the effectiveness of the bike-sharing program, enabling them to make informed decisions regarding investments in bike infrastructure and sustainable transportation initiatives.
- Visualization and Communication: Create compelling data visualizations and clear communication of findings to make the analysis accessible and easily understandable for various stakeholders.

2 Requirment Analysis

2.1 Functional Requirements

- Data Extraction and Preparation: Identify and select relevant datasets from Google Cloud's public dataset. Extract and preprocess the selected data to ensure it is in a suitable format for streaming.
- Real-time Streaming: Utilize Google Cloud Pub/Sub to enable the real-time streaming of prepared data. Publish the data onto Google Cloud Pub/Sub topics at a frequency that supports real-time processing.
- Streaming Analytics: Develop Python scripts to subscribe to the Pub/Sub topics and process incoming streaming data. Implement real-time analysis algorithms to identify patterns, trends, anomalies, or any other specific insights required.
- Data Visualization: Use visualization libraries (e.g., Matplotlib, Plotly) to create charts, graphs, and visual representations of the analyzed streaming data. Display the visualizations in a user-friendly format for easy interpretation.
- Scalability and Performance: Design the streaming architecture to accommodate varying data volumes without compromising performance. Implement best practices for optimizing data processing and analysis speed.

2.2 Data Security and Compliance Requirements

- Ensure that the selected datasets and the streaming process comply with data privacy regulations, if applicable.
- Implement proper access controls and authentication mechanisms to protect sensitive data during streaming and analysis.

2.3 Deliverables

The project is expected to deliver the following key outcomes:

- A Python-based streaming analytics solution capable of processing and analyzing incoming data from Google Cloud Pub/Sub in real time.
- Visualizations showcasing patterns, trends, or insights derived from the streaming data.
- Documentation detailing the project's architecture, setup instructions, and code explanations.
- An assessment of the solution's scalability and performance under different data loads.

3 Implementation

3.1 Data Extraction and Preparation

- Selection of Data: The project began by identifying and selecting relevant datasets from Google Cloud's public dataset repository. The chosen datasets aligned with the project's goals and objectives.
- Data Pre processing: Extracted data was preprocessed to ensure its compatibility with the streaming process. This involved cleaning, transforming, and structuring the data to facilitate real-time processing.

3.2 Real-time Streaming using Google Cloud Pub/Sub

- Setting up Pub/Sub: A Google Cloud Pub/Sub topic was created to serve as the entry point for the streaming data. Publishers were configured to send the preprocessed data to this topic.
- Publishing Data: Data from the selected dataset was programmatically published to the Pub/Sub topic. The publishing process was designed to ensure a steady flow of data into the streaming pipeline

3.3 Streaming Analytics using Python

- Subscribing to Pub/Sub Topic: Python scripts were developed to subscribe to the Pub/Sub topic. These scripts established a connection to the topic, enabling the retrieval of streaming data as it became available.[2]
- Real-time Analysis: The streaming data was ingested in real time using the Python scripts. Customized algorithms were implemented to analyze the data for patterns, trends, and insights. This analysis process occurred concurrently with the data streaming, ensuring real-time responsiveness.
- Data Enrichment: In some cases, additional data enrichment steps were performed during the analysis phase to enhance the quality and depth of insights obtained.

3.4 Data Visualization

- Visualization Libraries: Python's visualization libraries, such as Matplotlib and Plotly, were employed to create charts, graphs, and visual representations of the analyzed streaming data.
- Dashboard Creation: The visualizations were organized into intuitive dashboards that provided a comprehensive view of the streaming data insights. Users could interact with the visualizations to gain deeper understanding.

4 Workflow

4.1 Data Source Selection

- This is the step of choosing a relevant dataset from Google Cloud's public data repository. This dataset will serve as the source of data for your analysis.We are choosing google cloud austin bikeshare dataset as a source of data.

4.2 Python Script for Data Ingestion

- Write a Python script that fetches data from the chosen Google Cloud public dataset.
- Process the data as needed to ensure it's in a suitable format for analysis.

4.3 Publishing Data to Pubsub Topic

- Utilize Google Cloud Pub/Sub to enable real-time streaming of the data.
- Create a Pub/Sub topic to serve as the entry point for the streaming data.
- Modify your Python script to publish the processed data to the Pub/Sub topic.

4.4 Python Pipeline for Data Transformation

- Develop a Python pipeline that subscribes to the Pub/Sub topic and receives the streaming data.
- Implement necessary transformations on the incoming data. This could include cleaning, aggregation, filtering, and more.
- Leverage windowing concepts to group the data into logical time-based windows. This enables processing and analysis over specific time intervals

4.5 Storing Processed Data on BigQuery

- Set up a BigQuery dataset and table where you'll store the processed data.
- Configure the Python pipeline to insert the transformed data into the designated BigQuery table.
- The transformed data will be structured and ready for analysis within BigQuery.

4.6 Testing and Optimization

- Test the entire workflow with varying data volumes and streaming frequencies to ensure stability and performance.
- Optimize data processing and analysis steps for efficiency and speed.
- Address any bottlenecks or issues that arise during testing.

4.7 Interactivity and Insights

- Design the visualizations to provide insights into the streaming data.
- Allow users to interact with the visualizations to drill down into specific time frames or data points.
- The interactivity will help users gain a deeper understanding of the data trends and patterns.

Figure 1: Workflow Diagram

5 Experiments

5.1 Type of Bikes Used in Trips

After conducting an in-depth analysis of the Austin Bike Share data, a significant finding emerged: trips were predominantly made using one of two distinct types of bikes. The dataset revealed that out of all the recorded trips, a staggering 1,331,702 trips were completed using Classic bikes, while 660,024 trips were undertaken on Electric bikes. This clear demarcation between the two bike types underscores the preferences of users and sheds light on the patterns of bike utilization within the Austin Bike Share program

Table 1: Type of bike used in trips

Bike Type	No of Trips
Classic	1331702
Electric	660024

The implications of this discovery are profound and provide valuable insights into user behavior and bike share program optimization. The overwhelming preference for Classic bikes suggests that users might prefer a more traditional biking experience, potentially due to familiarity or comfort with this bike type. On the other hand, the substantial usage of Electric bikes indicates a growing interest in alternative transportation modes that offer convenience and reduced physical exertion. These results emphasize the importance of maintaining a diverse fleet of both Classic and Electric bikes to cater to the varying preferences and needs of users. Furthermore, this information can guide future infrastructure investments and resource allocation, enabling the bike share program to effectively meet the demands of its user base. Ultimately, the data-driven conclusion that the majority of trips are concentrated within these two bike types offers actionable insights to enhance the overall biking experience and promote sustainable urban mobility.

5.2 Bikes With Most Trips

Upon completion of a comprehensive analysis of the Austin Bike Share data, a notable revelation came to light: the identification of the top 10 bikes that have been used for the most trips. Among these, bike ID 559 stands out with a remarkable 3,816 trips attributed

to it, followed closely by bike IDs 951 and 593, each with 3,575 trips. Additionally, bike ID 107 and 371 have accumulated 3,563 and 3,543 trips, respectively. These findings underscore the significant role that these specific bikes play in the overall bike share program's usage patterns.

Table 2: Bike used for most trips in first week of April 2023

Bike Id	Trips
21490	77
21553	68
19040	64
21702	63
21550	62
21740	61
21698	58
22449	57
21480	56
19251	55

The implications of these top 10 bikes' usage patterns provide valuable insights into several aspects of the bike share program. First, the consistently high usage of these bikes suggests that they are particularly well-maintained and desirable among users. The fact that bike ID 559 has the highest number of trips could be attributed to its optimal condition, central location, or other factors that make it a preferred choice for many users. Moreover, the relatively equal number of trips for bike IDs 951 and 593 indicates that these bikes might be stationed at strategic locations or are popular among users for specific routes. Bike IDs 107 and 371, while slightly behind in trip count, still exhibit a remarkable usage rate, potentially owing to their efficient performance or favorable features. These results underline the importance of continuous maintenance and monitoring of the top-performing bikes to ensure a high-quality biking experience for users. Additionally, the data sheds light on potential strategies to optimize bike allocation and distribution across various stations to cater to the consistent demand for these popular bikes. In essence, the identification of the top 10 bikes used for the most trips not only highlights their contribution to the program's success but also offers actionable insights for enhancing user satisfaction and the overall effectiveness of the bike share system.

5.3 Most Used Bikes

Upon conducting a thorough analysis of the Austin Bike Share data, a significant revelation has come to light: the identification of the top 10 bikes that have been used for the maximum duration of time up to the present moment. Among these, bike ID 370 stands out with an impressive usage duration of 137,641 minutes, closely followed by bike IDs 893, 520, 135, and 920, which have been used for 130,749, 122,119, 120,607, and 115,076

minutes, respectively. This information offers valuable insights into the usage patterns and potential maintenance requirements of these bikes within the bike share program.

The extensive duration of use for these top 10 bikes points towards several noteworthy conclusions regarding their maintenance and other pertinent aspects. Firstly, the fact that bike ID 370 has accumulated the highest usage duration could imply that it has consistently provided a reliable and well-performing experience for users. Such bikes might be regularly serviced, ensuring they remain in optimal condition for extended periods. Similarly, the substantial usage durations of bike IDs 893, 520, 135, and 920 indicate their reliability and user preference, suggesting that these bikes are likely well-maintained and equipped to handle various terrains and conditions.

Table 3: Bike used for most time in first week of April 2023

Bike Id	Usage(Minutes)
213	21279
1687	13277
349G	6816
369G	2637
19821	1952
18995	1891
21476	1783
18181	1475
19748	1406
22573	1396

The maintenance requirements for these high-duration bikes could be relatively higher due to the prolonged wear and tear they undergo. Regular inspections, component replacements, and attention to structural integrity are crucial to ensuring these bikes continue to offer a safe and enjoyable riding experience. Furthermore, insights from these usage patterns can aid in strategically allocating maintenance resources to address the unique needs of these frequently used bikes. Additionally, this information can guide decisions related to retiring bikes that have served their purpose and replacing them with newer models to maintain a high standard of service.

Moreover, the usage duration data sheds light on user preferences and routes that these bikes are frequently chosen for. Bike ID 370, with the highest usage duration, might be stationed in a location with high footfall or could be preferred for specific routes known for their scenic views or convenience. This data, when combined with user feedback and other geographical factors, can help in optimizing bike allocation and station placement.

In essence, the revelation of the top 10 bikes used for the maximum duration of time not only highlights their reliability and user appeal but also informs the program's maintenance strategy, resource allocation, and overall user experience enhancement.

5.4 Busy Stations

After conducting a comprehensive analysis of the Austin Bike Share data, a notable discovery has emerged: the identification of the most frequently used stations in terms of trip counts. Among these, the station named '21st and Speedway @PC' with the station ID '3798' takes the lead with an astounding 533,865 trips associated with it. Following closely, '4th/Sabine' with station ID '2498' has been utilized 252,853 times, while 'Guadalupe and 21st' (station ID '2547'), 'Riverside @ S. Lamar' (station ID '2575'), and 'Nueces and 26th' (station ID '3838') have been selected for 198,027, 197,941, and 193,210 trips respectively. This information underscores the significant role that these stations play in shaping the travel patterns of the bike share program's users.

The insights derived from the frequency of station usage offer valuable considerations for optimizing the bike share program and enhancing user experiences. The prominence of '21st and Speedway @PC' as the most frequently used station suggests its strategic location within the city's network, making it a central hub for commuting and leisure activities. '4th/Sabine' follows suit, reflecting its convenience and popularity among riders for accessing key destinations. The high utilization of 'Guadalupe and 21st,' 'Riverside @ S. Lamar,' and 'Nueces and 26th' stations signifies their effectiveness in catering to user needs, whether for work, leisure, or connecting with other modes of transportation.

Table 4: Most busy stations in 1st week of April 2023

Station Id	Station Name	No of times Used
3798	21st and Speedway @PCL	1401
3794	Dean Keeton/Speedway	877
3838	26th/Nueces	560
2547	21st/Guadalupe	557
3841	28th/Rio Grande	520

Predictions can be drawn from this data to inform decision-making and improvements within the bike share program. Firstly, investing in expanding the capacity and infrastructure of these frequently used stations could alleviate potential issues of station congestion and bike shortages. Additionally, optimizing station placement based on the analysis of these heavily used locations could encourage ridership growth in surrounding areas.

Measures that can be taken based on this analysis include enhancing station maintenance and capacity management at these high-traffic locations. Ensuring that bikes are consistently available and well-maintained at these stations can enhance user satisfaction and encourage repeat usage. Moreover, the popularity of these stations may indicate certain routes and neighborhoods with high demand for biking options, which could inform urban planning and transportation initiatives.

In conclusion, the revelation of the most frequently used stations in Austin's bike

share program not only underscores their significance but also offers actionable insights for expanding station capacity, improving infrastructure, and strategically planning station locations. This analysis contributes to a more efficient and user-centric bike share experience while facilitating sustainable urban mobility solutions.

5.5 Busy Hours

Upon conducting a thorough analysis of the Austin Bike Share data, a significant pattern has emerged: the identification of the specific times of day when the most trips occur. The data reveals that at 17:00 (5:00 PM), a remarkable 167,702 trips take place, closely followed by 16:00 (4:00 PM) with 163,163 trips and 15:00 (3:00 PM) with 161,751 trips. Subsequently, the trend continues with 13:00 (1:00 PM) recording 159,832 trips, 14:00 (2:00 PM) witnessing 155,311 trips, and 12:00 (12:00 PM) registering 152,956 trips. These time-specific trip counts provide valuable insights into the temporal dynamics of bike share utilization.

The observations drawn from this temporal distribution offer compelling insights into user behavior and broader implications for optimizing the bike share program. The prominent surge in trips during the afternoon hours, particularly around 17:00, suggests a high demand for bikes during the evening commute hours. This could indicate that the bike share program serves as a convenient alternative for individuals to navigate their way home after work or school.

Table 5: Hourly data for 15/April/2023

Date	Hour of day	Trip count
15/Apr/23	13	96
15/Apr/23	14	89
15/Apr/23	11	89
15/Apr/23	15	81
15/Apr/23	19	81
15/Apr/23	17	79

Furthermore, the notable clustering of trips around midday (12:00 PM) and the early afternoon (1:00 PM and 2:00 PM) could indicate its popularity for lunchtime commutes, errands, or leisure rides. These findings underline the versatility of the bike share program, meeting the needs of users for both practical transportation and recreational purposes.

The implications of these temporal patterns extend beyond user preferences. Bike stations and resources can be strategically allocated to accommodate the higher demand during these peak hours. Additional bikes could be stationed near office complexes, educational institutions, and busy areas around these times to ensure availability and promote sustainable urban mobility.

In conclusion, the identification of the times of day with the highest trip occurrences provides valuable insights into user travel patterns and preferences. By recognizing the temporal trends of bike share utilization, the program can tailor its services, optimize resource allocation, and ultimately enhance the overall biking experience for a diverse range of users.

6 Code

6.1 Code for Publisher

```
from google.cloud import bigquery
from google.cloud import pubsub_v1
from datetime import datetime, timedelta
from utils.parser import setup_parser_publisher
from dotenv import dotenv_values
import random
import json
import time

env_config = dotenv_values(".env")


def sleep_random(max_sleep_seconds):
    try:
        rand_factor = random.randint(1,max_sleep_seconds)
        time.sleep(rand_factor)
        pass
    except Exception as e:
        print(f"Error sleeping:{e}")
        return


def generate_random_offset(offset_mins_limit):
    rand_factor = random.randint(-1, 1)
    offset_minutes = random.randint(0, offset_mins_limit)
    offset = rand_factor * timedelta(minutes=offset_minutes)
    return offset


def fetch_data_from_bigquery(data_source,rows_limit):
    try:
```

```
        bigquery_client = bigquery.Client()
        query = f"SELECT * FROM {data_source} LIMIT{rows_limit}"
        query_job = bigquery_client.query(query)
        rows = query_job.result()
        return rows
    except Exception as e:
        print(f"Error fetching data from BigQuery:{e}")
        return []


def transform_and_publish(
    rows, date_column,offset_mins_limit,output_topic,max_sleep_mins
):
    try:
        publisher = pubsub_v1.PublisherClient()
        for row in rows:
            row = dict(row.items())
            offset = generate_random_offset(offset_mins_limit)
            row[date_column] = (datetime.now()+offset).isoformat()
            row = json.dumps(row, indent=4)
            future = publisher.publish(output_topic,
            row.encode("utf-8"))
            future.result()
            print(row)
            sleep_random(max_sleep_mins)
            print("Published message to Pub/Sub topic.")
    except Exception as e:
        print(f"Error transforming and publishing data:{e}")
        return


if __name__ == "__main__":
    (known_args, _) = setup_parser_publisher()
    data_source = (
        known_args.data_source
        if known_args.data_source is not None
        else env_config.get("DATA_SOURCE")
    )
    date_column = (
        known_args.date_column
        if known_args.date_column is not None
        else env_config.get("DATE_COLUMN")
    )
```

```
    output_topic = (
        known_args.output_topic
        if known_args.output_topic is not None
        else env_config.get("OUTPUT_TOPIC")
    )
    rows_limit = (
        known_args.rows_limit
        if known_args.rows_limit is not None
        else int(env_config.get("ROWS_LIMIT") or 0)
    )
    max_offset_mins = (
        known_args.max_offset_mins
        if known_args.max_offset_mins is not None
        else int(env_config.get("MAX_OFFSET_MINS") or 0)
    )
    max_sleep_seconds = (
        known_args.max_sleep_seconds
        if known_args.max_sleep_seconds is not None
        else int(env_config.get("MAX_SLEEP_SECONDS") or 0)
    )
    if known_args.no_op:
        print("\nNo-op mode enabled. No data will be published.")
        print("Configuration values are as follows:")
        print(f"Data source:{data_source}")
        print(f"Date column to be transformed:{date_column}")
        print(f"Pub/Sub topic to publish to: {output_topic}")
        print(f"Number of rows to be queried: {rows_limit}")
        print(f"Maximum possible offset in mins: {max_offset_mins}")
        print(f"Maximum sleep time in seconds: {max_sleep_seconds}")
        exit()
    rows = fetch_data_from_bigquery(data_source, rows_limit)
    transform_and_publish(
        rows, date_column, max_offset_mins, output_topic,
        max_sleep_seconds
    )
```

6.2 Code snippet of main pipeline

```
import os
import sys
import json
import apache_beam as beam
```

```
from datetime import datetime
from apache_beam.options.pipeline_options import SetupOptions
from apache_beam.options.pipeline_options import PipelineOptions
from apache_beam.options.pipeline_options import StandardOptions
from app.subscriber_type import SubscriberTypeCount
from app.subscriber_type import FormatSubscriberType
from app.bigquery import WriteToBigQuery
from utils.logger import setup_logger


class ParseTimestamp(beam.DoFn):
    """Parses the timestamp from the input
    string into a timestamp object.
    The string is assumed to be formatted
    as ISO 8601.For example: 2019-01-14T12:34:56.000Z
    """

    def process(self, element):
        ts = element["start_time"]
        element["start_time"] = datetime.fromisoformat(ts).
        timestamp()
        yield element


    def run(pipeline_options, logger, input_topic,
        bigquery_dataset, project):
        """Runs the pipeline."""
        try:
            pipeline = beam.Pipeline(options=pipeline_options)
            data = (
                pipeline
                | "Read from PubSub" >> beam.io.ReadFromPubSub(
                   topic=input_topic)
                | "Parse data" >> beam.Map(lambda elem: json.loads(
                   elem))
                | "Parse timestamp" >> beam.ParDo(ParseTimestamp())
                | "Add timestamp"
                >> beam.Map(
                    lambda elem: beam.window.TimestampedValue(
                        elem, elem["start_time"]
                    )
                )
            )
```

```
        subscriber_type_count = (
            data
            | "Extract and find subscriber type count" >>
               SubscriberTypeCount()
            | "Format subscriber type count" >>
               beam.ParDo(FormatSubscriberType())
            | "Write results to BigQuery"
            >> WriteToBigQuery(
                "subscriber_type_count",
                bigquery_dataset,
                {
                    "window_start":"STRING",
                    "window_end":"STRING",
                    "subscriber_type":"STRING",
                    "count": "INTEGER",
                    "processing_time":"STRING",
                },
                project,
            )
        )

        result = pipeline.run()
        result.wait_until_finish()
    except KeyboardInterrupt:
        logger.warning("Interrupted.")
        logger.info("Stopping pipeline")
        try:
            sys.exit(0)
        except SystemExit:
            os._exit(0)


def main(known_args, options):
    logger = setup_logger()
    logger.info("starting app")

    pipeline_options = PipelineOptions(options)
    pipeline_options.view_as(SetupOptions).save_main_session = True
    pipeline_options.view_as(StandardOptions).streaming = True

    input_topic = known_args.input_topic
    project_id = known_args.project_id
```

```
    bigquery_dataset = known_args.bigquery_dataset

    if known_args.no_op:
        print("\nNo-op mode enabled. No data will
        be read and transformed.")
        print("Configuration values are as follows:")
        print(f"Input Pub/Sub topic:{input_topic}")
        print(f"Project ID:{project_id}\n")
        print(f"Output BigQuery dataset:{bigquery_dataset}")
        sys.exit(0)

    run(pipeline_options, logger, input_topic,
    bigquery_dataset, project_id)
    logger.info("exiting app...")
```

6.3 A helper class which contains the logic to translate the user's custom dict into a BigQuery table row and then write it to BigQuery

```
import apache_beam as beam
class WriteToBigQuery(beam.PTransform):
    """Generate, format, and write BigQuery table row
    information."""

    def __init__(self, table_name, dataset, schema, project):
        """Initializes the transform.
        Args:
            table_name:Name of the BigQuery table
            to use.
            dataset: Name of the dataset to use.
            schema:Dictionary in the format
            {'column_name':bigquery_type'}
            project: Name of the Cloud project
            containing BigQuery table.
        """
        beam.PTransform.__init__(self)
        self.table_name = table_name
        self.dataset = dataset
        self.schema = schema
        self.project = project

    def get_schema(self):
```

```
        """Build the output table schema."""
        return ", ".join("%s:%s" % (col, self.schema[col])
        for col in self.schema)

    def expand(self, pcoll):
        return (
            pcoll
            | "Convert to row"
            >> beam.Map(lambda elem: {col: elem[col]
            for col in self.schema})
            | beam.io.WriteToBigQuery(
                self.table_name, self.dataset, self.project,
                self.get_schema()
            )
        )
```

6.4 This code contains a pipeline that calculates the number of trips by subscriber type in a given time window

```
import apache_beam as beam
from datetime import datetime

class SubscriberTypeCount(beam.PTransform):
    """Extracts the subscriber type from the input dictionary
    and outputs a tuple of the subscriber type and a count of 1.
    The count can then be aggregated to find the total number of
    trips by a subscriber type.
    """

    def __init__(self):
        beam.PTransform.__init__(self)
        self.window_duration = 1 * 60  # 1 minute

    def expand(self, pcoll):
        return (
            pcoll
            | "Window into fixed windows"
            >> beam.WindowInto(
                beam.window.FixedWindows(self.window_duration)
            )
```

```
            | "Extract subscriber types"
            >> beam.Map(lambda elem: (elem["subscriber_type"], 1))
            | "Count subscriber types" >> beam.CombinePerKey(sum)
        )


class FormatSubscriberType(beam.DoFn):
    """Formats the subscriber type count into a dictionary.
    """

    def process(self, element, window=beam.DoFn.WindowParam):
        (subscriber_type, count) = element
        start = window.start.to_utc_datetime().isoformat()
        end = window.end.to_utc_datetime().isoformat()
        yield {
            "window_start": start,
            "window_end": end,
            "subscriber_type": subscriber_type,
            "count": count,
            "processing_time": datetime.utcnow().isoformat(),
        }
```

6.5 Github link to complete code base developed and maintained actively

https://github.com/7ze/austin-bikeshare-data-analysish

7 Discussion

In the course of this project, multiple goals were pursued to enhance understanding and decision-making. Notably, the project successfully achieved the generation of insightful findings from the available data, shedding light on underlying patterns and trends. Optimization opportunities within the existing processes were also identified, potentially leading to increased efficiency. The goal of accurately forecasting demand was met, providing a glimpse into future resource needs. Additionally, user profiling was accomplished, aiding in tailoring services to user preferences. Performance evaluation was also undertaken, allowing for a comprehensive assessment of relevant metrics.

However, certain goals were not attained. The objective of conducting a seasonal and geographic analysis could not be fulfilled, likely due to limitations in the data available for these specific dimensions. Similarly, the goal of conducting a comparative analysis

could not be realized, possibly due to a lack of appropriate benchmarks or comparable data sets. These unachieved goals highlight the importance of robust and comprehensive data collection to enable meaningful analysis.

In preliminary conclusion, the project demonstrated substantial success in delivering valuable insights, optimization avenues, demand forecasting, user understanding, and performance assessment. Despite certain unmet goals, the accomplishments made have the potential to significantly impact decision-making processes. It is recommended that future efforts focus on enhancing data collection strategies and expanding the scope of analysis to encompass previously unaddressed dimensions, thus providing a more holistic view for more informed decision-making.

8 Conclusion

The Austin Bike Share Data Analysis project has successfully achieved its primary goals while also revealing certain areas for potential improvement. Valuable insights were generated from the available data, leading to the identification of optimization opportunities within the bike share system. Accurate demand forecasting was accomplished, enabling better resource allocation and planning. User profiling facilitated tailored services to meet user preferences, and a comprehensive performance evaluation shed light on the system's efficiency.

There are several steps which could be taken to further enhance its impact. First, efforts should be focused on improving data collection strategies, especially regarding seasonal and geographic data. This would allow for a more comprehensive analysis that could uncover regional trends and seasonal fluctuations. Second, dedicating time to establish meaningful benchmarks and comparable data points would facilitate the much-needed comparative analysis.

Additionally, exploring external data sources and expanding the scope of analysis to encompass a broader range of factors could enrich the project's findings. Lastly, considering the dynamic nature of bike-sharing systems, devoting resources to real-time data integration and predictive modeling could provide actionable insights for more agile decision-making.

In conclusion, the Austin Bike Share Data Analysis project has delivered significant value through accomplished goals while revealing the importance of robust data collection for comprehensive analysis. The project's insights have the potential to drive strategic decisions, and with additional time, a more holistic approach could further amplify its impact.

References

[1] Junfeng Jiao and Shunhua Bai. Understanding the shared e-scooter travels in austin, tx. *International Journal of Geo-Information*, 9:1–12, 02 2020.

[2] Apache Beam. Design your pipeline - apache beam documentation, 2023.

Eidesstattliche Erklärung

I hereby affirm that I have written this paper independently and have not used any sources or aids other than those indicated, and that I have clearly marked the citations. I further declare that the present work has not yet been submitted in the same or similar form in the context of another other examination procedure.

Fulda, the September 2, 2023 Shinu Donney Usama Tahir