CS432/532: Final Project Report

Hotel Customer Dataset Analysis and Revenue Prediction using NoSQL Database

Team Member(s): Karthik Shanmugam, Akash Rasal

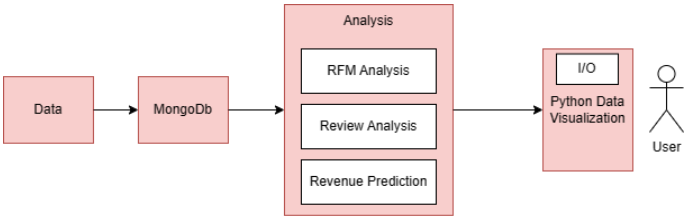
I. Problem

While working in a competitive hospitality industry, optimizing customer satisfaction, and maximizing revenue are very important for sustained business success. It is important for the hotel business to come up with new strategies to attract more customers and address existing issues in their business. With aim to facilitate strategy making and assist business planning, we analyze a hotel customer dataset. The dataset contains information about every customer of the hotel. For every customer, the dataset contains attributes of nationality, age, average lead time, lodging revenue, other revenue, bookings cancelled, no-show, checked-in, days of last stay, customer review and more fields. We have individuals from 188 unique nationalities, and reviews as text.

We have performed RFM analysis (Recency, Frequency and Monetary analysis) on the data using the DaysSinceLastStay, AverageLeadTime, Bookings Information, LodgingRevenue and OtherRevenue. Using the RFM analysis, we are able to segment customers into 5 different segments which can be used to analyze most valued customers and least valued customers. We analyze the customer reviews for positive and negative comments. Using marked reviews, we can see which word is used the most and we can take actions accordingly. To mark the review, check for positive and negative words in a review and weigh positive and negative words against each other and mark the review accordingly. Lastly, we have created a naïve revenue predictor over lodging revenue using average revenue by month and using Simple Moving Average for 6 data points.

II. Software Design and Implementation

A. *Software Design and NoSQL-Database and Tools Used*



The above diagram shows the system design used for analyzing the data and generating significant results. Initially, all the data is cleaned for incorrect attributes, null values, and missing values. The filtered data was imported in MongoDB.

In the data analysis stage, three different analyses were performed, RFM analysis, Customer review analysis, and Total revenue prediction. All the analyses were done using MongoDB queries and pipelines and we have used $merge, $lookup and similar operators of MongoDB.

In the last stage, the analyses from MongoDB pipelines were integrated with Python and plotting libraries, different plots were created for the analyses. For all the plots, we have used Plotly library in python. To create a web page to show the analyses, we used streamlit library to create a web application and display plots.

B. *Parts that you have implemented*

Task 1: RFM Analysis (implemented by Karthik Shanmugam)

An analytical approach known as RFM analysis was applied to gain insights into customer behavior. Recency, Frequency, and Monetization were considered, and respective attributes, d_Recency, d_Frequency, and d_Monetization, were derived and weighted and minimized or maximized appropriately to formulate a composite metric termed CustomerRank. Using CustomerRank, the customer base was divided into five groups/quartiles, each with a unique SegmentRank. Various attributes, such as lodging revenue, lead time and number of nights spent, were analyzed to identify the most valuable customer segments. The goal of this analysis is to identify the most valuable customer segment providing insights for decision making. Advertising to potential new customers is a critical function of maximizing revenue and looking at the distribution channel (advertising medium) used for the top segment of customers (SegmentRank-1) provides insight into the best distribution channel that can be used for targeted marketing.

Task 2: Revenue prediction (implemented by Akash Rasal)

The revenue prediction includes a systematic approach using the various MongoDB pipelines. First, we established a pipeline to calculate average monthly revenue using historical data using lodging revenue and other revenue attributes. All the generated data was stored in a MongoDB collection. Another pipeline was created, which uses simple moving average (SMA) over the past 6-month period from current month to forecast real-time earnings. Using lookups and comparing the current calculated data with the initial revenue estimate is done to increase accuracy which creates a mixture of historical insights.

Task 3: Customer Review Analysis (implemented by Karthik Shanmugam and Akash Rasal)

For analyzing sentiments of customer reviews for the hotel, a MongoDB pipeline was used to systematically extract and count words. This data is used in subsequent pipelines. Using a pre-existing dataset of positive and negative words, each word is classified as positive or negative, facilitating a basic understanding of sentiment of the word. Using these classifications, another MongoDB pipeline is used to categorize the whole review as positive or negative. To classify the review as positive or negative, we weighed the positive and negative words in the review and then classify the review accordingly. Using additional customer details, such as nationality, the analysis provides valuable insights into how different countries perceive the hotel experience.

III. Project Outcome

In this project we analyzed the hotel customer dataset for its customer segmentation, customer reviews by nationality and predicted revenue for the next six months. The RFM analysis segments the customers into different segments using which we can target the customers to visit the hotel again. The marketing team can use this analysis to target the most valuable customers and the average valuable customers. It can go hand in hand with the nationality of the most valuable customers. Classifying the reviews into positive and negative gives a broad view of how the customers think about the hotel. Using review segmentation by country, the hotel can ensure that the service is improved for the customers of a particular nationality. The revenue prediction helps to gauge the incoming revenue over next six months. Using this predictor, the analyzer can focus on ways to increase the revenue, and any upcoming holiday season and manage his staff accordingly.

i. RFM Analysis

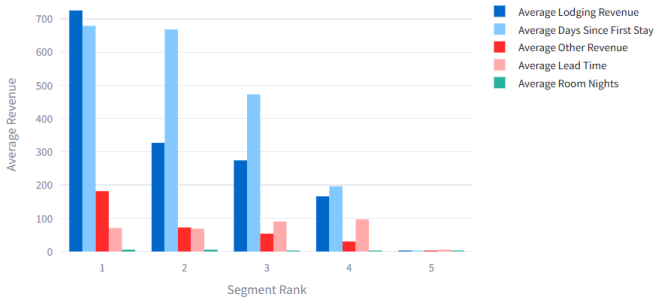
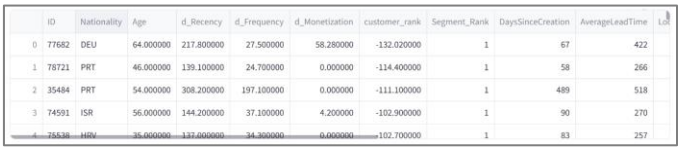


Fig 1. Customer segmentation by RMF analysis



	ID	Nationality	Age	d_Recency	d_Frequency	d_Monetization	customer_rank	Segment_Rank	DaysSinceCreation	AverageLeadTime	La
0	77682	DEU	64.000000	217.800000	27.500000	58.280000	-132.020000	1	67	422	
1	78721	PRT	46.000000	139.100000	24.700000	0.000000	-114.400000	1	58	266	
2	35484	PRT	54.000000	308.200000	197.100000	0.000000	-111.100000	1	489	518	
3	74591	ISR	56.000000	144.200000	37.100000	4.200000	-102.900000	1	90	270	
4	75538	HRV	35.000000	137.000000	34.300000	0.000000	-102.700000	1	83	257	

Fig 2. Customer Rank and Calculated metrics by RMF analysis

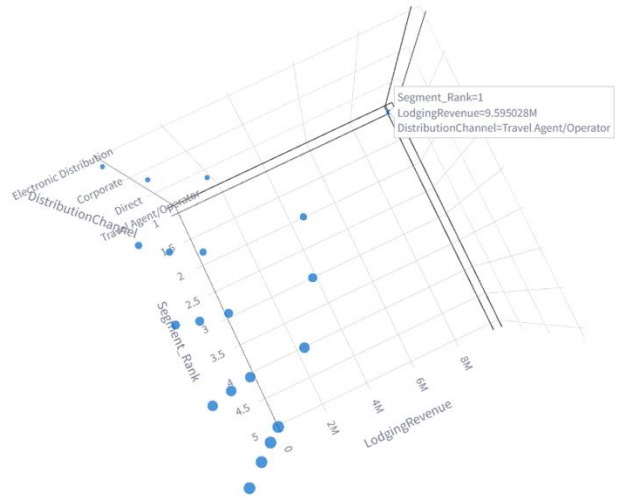


Fig 3. 3-D plot of Lodging revenue, Segment Rank and Distribution channel(Segment 1 has highest revenue and distribution channel is Travel Agent)

Fig. 1, shows the most valuable customers in segment 1, second most valuable customers in segment 2 and similar for other segments. As we can understand, segment 1 customers generate the most revenue at about $900 per customer. We can also see that these are older customers as they first visited the hotel 650 days ago. Also, they have very low lead time, means they book for closer dates in the future. Similarly, we can see that segment 4 customers seem to be some recent customers, and, on average, they spend less at the hotel. Using this analysis, the marketing team can focus on segment 2 and 3 to send some promotional offers and revisit the changes in the recent past for more customers in segment 4. The tabular fig. 2 helps to get more data about the customer from their customer rank facilitating easy retrieval of data.

ii. Revenue prediction

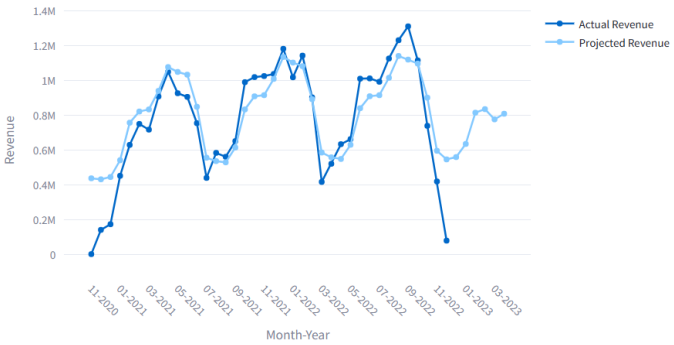


Fig 4. Actual vs Projected revenue

Fig. 4 shows actual revenue vs the projected revenue. It is a time-series graph which has date on the X-axis and revenue on the Y-axis. From the plot, we can observe that the projected revenue is accurate to the actual data, and this asserts that the projection is valid. The projected revenue for the next six months will help in assisting planning accordingly. We can also

observe that the revenue grows in the Summer season and the projected data shows the same, a rising edge for coming Summer season.

iii. Customer Review Analysis

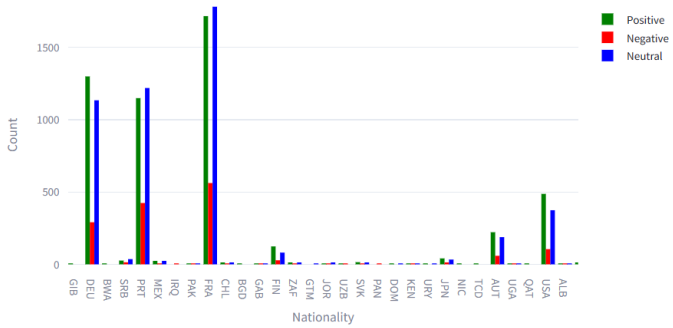


Fig 5. Geospatial Customer review analysis

Fig 6. WordCloud of Most used words in reviews

Fig 5 shows a plot of nationalities on the X-axis and review count on the Y-axis. From this, we can understand that France (FRA), Portugal (PRT) and German (DEU) nationals have reviewed the hotel the most and most of the reviews are neutral or positive. Also, we can see that customers with USA nationality have higher positive reviews. This affirms that customers like the hospitality of the hotel and might be willing to revisit and refer the hotel to their friends. We can also see that there are very few reviews from customers of other nationalities and can focus on attracting more customers from those nationalities. Fig. 6 shows the most used words which can help use identify the points which are good or needs improvement.

REFERENCES

[1] Dataset: Hotel Customer Dataset | Kaggle

[2] MongoDB Compass: MongoDB Compass | MongoDB

[3] MongoDB Time Series Data | MongoDB

[4] Aggregation Operations — MongoDB Manual

[5] Plotly library: Plotly Python Graphing Library

[6] Streamlit: Streamlit • A faster way to build and share data apps .

Code for RFM analysis pipelines:

```
import streamlit as st
from datetime import datetime, tzinfo, timezone
import pandas as pd
from pymongo import MongoClient
import plotly.express as px
import plotly.graph_objects as go

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import pandas as pd
import numpy as np

# Page configurations
PAGE_CONFIG = {"page_title": "RFM Analysis", "page_icon": ":pick:"}

# Set page title and icon
st.set_page_config(layout="wide", page_title=PAGE_CONFIG["page_title"],
page_icon=PAGE_CONFIG["page_icon"])

# Sidebar menu
# menu_options = ["Page 1", "Page 2", "Page 3"]
# selected_page = st.sidebar.radio("Select a Page", menu_options)

# Connect to MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['cs532']['HotelWithReviews']

total_customers = num_documents = db.count_documents({});

#-------------------------------------------------------------------------------------------
-----------------------------------------------
# Code for Data Table
#-------------------------------------------------------------------------------------------
-----------------------------------------------
pipeline = [
    {
        "$addFields": {
            "d_Recency": {
                "$add": [
                    {"$multiply": [0.1, {"$toInt": "$DaysSinceLastStay"}]},
                    {"$multiply": [0.5, {"$toInt": "$AverageLeadTime"}]}
                ]
            },
            "d_Frequency": {
```

```
            "$add": [
                {"$multiply": [0.4, {"$toInt": "$DaysSinceFirstStay"}]},
                {"$multiply": [0.5, {"$toInt": "$BookingsCanceled"}]},
                {"$multiply": [0.3, {"$toInt": "$BookingsCheckedIn"}]},
                {"$multiply": [0.35, {"$toInt": "$BookingsNoShowed"}]}
            ]
        },
        "d_Monetization": {
            "$add": [
                {"$multiply": [0.4, {"$toDouble": "$LodgingRevenue"}]},
                {"$multiply": [0.7, {"$toDouble": "$OtherRevenue"}]}
            ]
        },
        "customer_rank": {
            "$add": [
                {"$multiply": [0.6, {"$subtract": [1, "$d_Recency"]}]},
                {"$multiply": [0.2, "$d_Frequency"]},
                {"$multiply": [0.2, "$d_Monetization"]}
            ]
        }
    }
},
{
    "$addFields": {
        "customer_rank": {
            "$add": [
                {"$multiply": [1, "$d_Frequency"]},
                {"$multiply": [1, "$d_Monetization"]},
                {"$multiply": [-1, "$d_Recency"]}
            ]
        }
    }
},
{
    "$sort": {"customer_rank": 1}
},
{
    "$setWindowFields": {
        "sortBy": {"customer_rank": 1},
        "output": {"sorted_row_number": {"$documentNumber": {}}}
    }
},
{
    "$addFields": {
        "Segment_Rank": {
            "$switch": {
```

```
                    "branches": [
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.2,
total_customers]}]}, "then": 1},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.4,
total_customers]}]}, "then": 2},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.6,
total_customers]}]}, "then": 3},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.8,
total_customers]}]}, "then": 4},
                        {"case": "True", "then": 5}
                    ]
                }
            }
        }
    },
    {
        "$project": {
            "DocIDHash": 0,
            "NameHash": 0
        }
    }
]

# Execute the pipeline and get the result
result = list(db.aggregate(pipeline))

COLUMNS = [
    "ID", "Nationality", "Age", "d_Recency", "d_Frequency", "d_Monetization",
"customer_rank", "Segment_Rank",
    "DaysSinceCreation", "AverageLeadTime",
    "LodgingRevenue", "OtherRevenue", "BookingsCanceled", "BookingsNoShowed",
"BookingsCheckedIn",
    "PersonsNights", "RoomNights", "DaysSinceLastStay", "DaysSinceFirstStay",
"DistributionChannel",
    "MarketSegment", "SRHighFloor", "SRLowFloor", "SRAccessibleRoom", "SRMediumFloor",
"SRBathtub",
    "SRShower", "SRCrib", "SRKingSizeBed", "SRTwinBed", "SRNearElevator",
"SRAwayFromElevator",
    "SRNoAlcoholInMiniBar", "SRQuietRoom", "CustomerReview",
]

# Convert the result to a Pandas DataFrame
df = pd.DataFrame(result, columns=COLUMNS)
df = df.dropna()
# df_sorted = df.sort_values(by="Segment_Rank")
```

```
# filtered_df = df[df['Segment_Rank'] == 2]

# Display the first 1000 rows in a Streamlit table
st.title("RFM Segmentation Analysis Data")


st.dataframe(df.head(1000).style.set_table_styles([{
    'selector': 'thead th',
    'props': [('background-color', '#48D1CC'), ('color', 'white')]
}]), None, 200, use_container_width=True)


#-------------------------------------------------------------------------------------------------
# Code for Plot
#-------------------------------------------------------------------------------------------------

st.title("RFM Segmentation Analysis Plot")

pipeline_plot = [
    {
        "$addFields": {
            "d_Recency": {
                "$add": [
                    {"$multiply": [0.1, {"$toInt": "$DaysSinceLastStay"}]},
                    {"$multiply": [0.5, {"$toInt": "$AverageLeadTime"}]}
                ]
            },
            "d_Frequency": {
                "$add": [
                    {"$multiply": [0.4, {"$toInt": "$DaysSinceFirstStay"}]},
                    {"$multiply": [0.5, {"$toInt": "$BookingsCanceled"}]},
                    {"$multiply": [0.3, {"$toInt": "$BookingsCheckedIn"}]},
                    {"$multiply": [0.35, {"$toInt": "$BookingsNoShowed"}]}
                ]
            },
            "d_Monetization": {
                "$add": [
                    {"$multiply": [0.4, {"$toDouble": "$LodgingRevenue"}]},
                    {"$multiply": [0.7, {"$toDouble": "$OtherRevenue"}]}
                ]
            },
            "customer_rank": {
                "$add": [
                    {"$multiply": [0.6, {"$subtract": [1, "$d_Recency"]}]},
                    {"$multiply": [0.2, "$d_Frequency"]},
```

```
                    {"$multiply": [0.2, "$d_Monetization"]}
                ]
            }
        }
    },
    {
        "$addFields": {
            "customer_rank": {
                "$add": [
                    {"$multiply": [1, "$d_Frequency"]},
                    {"$multiply": [1, "$d_Monetization"]},
                    {"$multiply": [-1, "$d_Recency"]}
                ]
            }
        }
    },
    {
        "$sort": {"customer_rank": -1}
    },
    {
        "$setWindowFields": {
            "sortBy": {"customer_rank": -1},
            "output": {"sorted_row_number": {"$documentNumber": {}}}
        }
    },
    {
        "$addFields": {
            "Segment_Rank": {
                "$switch": {
                    "branches": [
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.2,
total_customers]}]}, "then": 1},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.4,
total_customers]}]}, "then": 2},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.6,
total_customers]}]}, "then": 3},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.8,
total_customers]}]}, "then": 4},
                        {"case": "True", "then": 5}
                    ]
                }
            }
        }
    },
    {
        "$group": {
```

```
            "_id": "$Segment_Rank",
            "averageLodgingRevenue": {"$avg": "$LodgingRevenue"},
            "averageOtherRevenue" : {"$avg": "$OtherRevenue"},
            "averageLeadTime": {"$avg":"$AverageLeadTime"},
            "averageDaysSinceFirstStay": {"$avg":"$DaysSinceFirstStay"},
            "averageRoomNights": {"$avg":"$RoomNights"}

        }
    },
    {
        "$project": {
            "DocIDHash": 0,
            "NameHash": 0
        }
    }
]

# Execute the pipeline and get the result
result_plot = list(db.aggregate(pipeline_plot))

# print(result_plot)
# Convert the MongoDB result to a Pandas DataFrame
df_plot = pd.DataFrame(result_plot)


# Display the original data
# st.subheader("Original Data:")
# st.write(df_plot)

# Plot the bar graph using Plotly and display it in Streamlit
# fig_plot = px.bar(df_plot, x="_id", y=["averageLodgingRevenue","averageOtherRevenue"],
labels={"_id": "Segment Rank", "averageLodgingRevenue": "Average Lodging Revenue",
"averageOtherRevenue": "Average Other Revenue"})
# fig_plot.update_layout(title="Average Lodging Revenue by Segment Rank",
xaxis_title="Segment Rank", yaxis_title="Average Revenue")
# st.plotly_chart(fig_plot)

fig_plot = go.Figure()

fig_plot.add_trace(go.Bar(x=df_plot["_id"], y=df_plot["averageLodgingRevenue"],
name="Average Lodging Revenue"))
fig_plot.add_trace(go.Bar(x=df_plot["_id"], y=df_plot["averageDaysSinceFirstStay"],
name="Average Days Since First Stay"))
fig_plot.add_trace(go.Bar(x=df_plot["_id"], y=df_plot["averageOtherRevenue"],
name="Average Other Revenue"))
```

```
fig_plot.add_trace(go.Bar(x=df_plot["_id"], y=df_plot["averageLeadTime"], name="Average
Lead Time"))
fig_plot.add_trace(go.Bar(x=df_plot["_id"], y=df_plot["averageRoomNights"], name="Average
Room Nights"))

fig_plot.update_layout(
    title="Average Revenue by Segment Rank",
    xaxis_title="Segment Rank", yaxis_title="Average Revenue",
    scene=dict(
        xaxis=dict(title="Segment Rank"),
        yaxis=dict(title="Revenue Type"),
        zaxis=dict(title="Average Revenue"),
    )
)

st.plotly_chart(fig_plot)

#-------------------------------------------------------------------------------------------
-------------------------------------------------
# Code for 3-D Plot
#-------------------------------------------------------------------------------------------
-------------------------------------------------
st.title("RFM Segmentation Analysis 3D Plot")

pipeline_plot_3d = [
    {
        "$addFields": {
            "d_Recency": {
                "$add": [
                    {"$multiply": [0.1, {"$toInt": "$DaysSinceLastStay"}]},
                    {"$multiply": [0.5, {"$toInt": "$AverageLeadTime"}]}
                ]
            },
            "d_Frequency": {
                "$add": [
                    {"$multiply": [0.4, {"$toInt": "$DaysSinceFirstStay"}]},
                    {"$multiply": [0.5, {"$toInt": "$BookingsCanceled"}]},
                    {"$multiply": [0.3, {"$toInt": "$BookingsCheckedIn"}]},
                    {"$multiply": [0.35, {"$toInt": "$BookingsNoShowed"}]}
                ]
            },
            "d_Monetization": {
                "$add": [
                    {"$multiply": [0.4, {"$toDouble": "$LodgingRevenue"}]},
                    {"$multiply": [0.7, {"$toDouble": "$OtherRevenue"}]}
                ]
```

```
            },
            "customer_rank": {
                "$add": [
                    {"$multiply": [0.6, {"$subtract": [1, "$d_Recency"]}]},
                    {"$multiply": [0.2, "$d_Frequency"]},
                    {"$multiply": [0.2, "$d_Monetization"]}
                ]
            }
        }
    },
    {
        "$addFields": {
            "customer_rank": {
                "$add": [
                    {"$multiply": [1, "$d_Frequency"]},
                    {"$multiply": [1, "$d_Monetization"]},
                    {"$multiply": [-1, "$d_Recency"]}
                ]
            }
        }
    },
    {
        "$sort": {"customer_rank": -1}
    },
    {
        "$setWindowFields": {
            "sortBy": {"customer_rank": -1},
            "output": {"sorted_row_number": {"$documentNumber": {}}}
        }
    },
    {
        "$addFields": {
            "Segment_Rank": {
                "$switch": {
                    "branches": [
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.2,
total_customers]}]}, "then": 1},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.4,
total_customers]}]}, "then": 2},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.6,
total_customers]}]}, "then": 3},
                        {"case": {"$lte": ["$sorted_row_number", {"$multiply": [0.8,
total_customers]}]}, "then": 4},
                        {"case": "True", "then": 5}
                    ]
                }
```

```
            }
        }
    },
    {
        "$group": {

            "_id": {"Segment_Rank": "$Segment_Rank", "DistributionChannel":
"$DistributionChannel"},
            "lodgingRevenuePerChannel": {"$sum": "$LodgingRevenue"}

        }
    },
    {
        "$project": {
            "Segment_Rank": "$_id.Segment_Rank",
            "LodgingRevenue":"$lodgingRevenuePerChannel",
            "DistributionChannel":"$_id.DistributionChannel"
        }
    }
]

# Execute the pipeline and get the result
result_plot_3d = list(db.aggregate(pipeline_plot_3d))


# Convert the MongoDB result to a Pandas DataFrame
df_plot_3d= pd.DataFrame(result_plot_3d)

fig_3d = px.scatter_3d(df_plot_3d,
                       x='Segment_Rank',
                       y='LodgingRevenue',
                       z='DistributionChannel',
                       size='Segment_Rank',
                      #   color='Segment_Rank',
                       opacity=0.7,
                       labels={'Segment_Rank': 'Segment_Rank'},
                       height=750,
                       width=1000)


# Streamlit app
# st.title('3D Plot for Segment Rank, Lodging Revenue, and Distribution Channel')
st.plotly_chart(fig_3d)
```

ii. Code for Predicting revenue

```
import streamlit as st
from datetime import datetime, tzinfo, timezone
import pandas as pd
from pymongo import MongoClient
import plotly.graph_objects as go
import streamlit as st
import matplotlib.pyplot as plt
import plotly.express as px

# Page configurations
PAGE_CONFIG = {"page_title": "SMA Analysis", "page_icon": ":brain"}

# Set page title and icon
st.set_page_config(layout="wide", page_title=PAGE_CONFIG["page_title"],
page_icon=PAGE_CONFIG["page_icon"])

# Requires the PyMongo package.
# https://api.mongodb.com/python/current

client = MongoClient('mongodb://localhost:27017/')

collection = client['cs532']['HotelWithReviews']


#Get top nationalities which stay at the hotel
topSpendingNationalityQuery = [
    {
        '$match': {
            'BookingsCheckedIn': {
                '$eq': 1
            }
        }
    }, {
        '$addFields': {
            'CalculatedMonth': {
                '$month': {
                    '$subtract': [
                        {
                            '$toDate': datetime.utcnow()
                        }, {
                            '$multiply': [
                                '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                            ]
                        }
                    ]
```

```
            }
        },
        'CalculatedYear': {
            '$year': {
                '$subtract': [
                    {
                        '$toDate': datetime.utcnow()
                    }, {
                        '$multiply': [
                            '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                        ]
                    }
                ]
            }
        },
        'calculatedDate': {
            '$subtract': [
                {
                    '$toDate': datetime.utcnow()
                }, {
                    '$multiply': [
                        '$DaysSinceCreation', 24 * 60 * 60 * 1000
                    ]
                }
            ]
        }
    }
}, {
    '$project': {
        '_id': 1,
        'CalculatedMonth': '$CalculatedMonth',
        'Nationality': '$Nationality',
        'LodgingRevenue': '$LodgingRevenue',
        'OtherRevenue': '$OtherRevenue'
    }
}, {
    '$group': {
        '_id': {
            'CalculatedMonth': '$CalculatedMonth',
            'Nationality': '$Nationality'
        },
        'count': {
            '$sum': 1
        },
        'LodgingRevenueForMonth': {
            '$sum': '$LodgingRevenue'
```

```
            },
            'OtherRevenueForMonth': {
                '$sum': '$OtherRevenue'
            }
        }
    }, {
        '$addFields': {
            'TotalRevenueForMonth': {
                '$add': [
                    '$OtherRevenueForMonth', '$LodgingRevenueForMonth'
                ]
            }
        }
    }, {
        '$sort': {
            '_id.CalculatedMonth': 1,
            'TotalRevenueForMonth': -1
        }
    }, {
        '$group': {
            '_id': '$_id.CalculatedMonth',
            'topNationalities': {
                '$push': '$_id.Nationality'
            },
            'topSpends': {
                '$push': '$TotalRevenueForMonth'
            }
        }
    }, {
        '$sort': {
            '_id': 1
        }
    }, {
        '$project': {
            '_id': 0,
            'month': '$_id',
            'topNationalities': {
                '$slice': [
                    '$topNationalities', 2
                ]
            },
            'topSpending': {
                '$slice': [
                    '$topSpends', 2
                ]
            }
```

```
        }
    }
]

topSpendingNationality = list(collection.aggregate(topSpendingNationalityQuery))



#Get Training Data: Group by revenue and store in db
trainDataMontlyPipeline = [
    {
        '$addFields': {
            'CalculatedMonth': {
                '$month': {
                    '$subtract': [
                        {
                            '$toDate': datetime.utcnow()
                        }, {
                            '$multiply': [
                                '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                            ]
                        }
                    ]
                }
            },
            'CalculatedYear': {
                '$year': {
                    '$subtract': [
                        {
                            '$toDate': datetime.utcnow()
                        }, {
                            '$multiply': [
                                '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                            ]
                        }
                    ]
                }
            },
            'calculatedDate': {
                '$subtract': [
                    {
                        '$toDate': datetime.utcnow()
                    }, {
                        '$multiply': [
                            '$DaysSinceCreation', 24 * 60 * 60 * 1000
```

```
                    ]
                }
            ]
        }
    }
}, {
    '$match': {
        'BookingsCheckedIn': {
            '$gt': 0
        }
    }
}, {
    '$project': {
        '_id': 1,
        'CalculatedMonth': '$CalculatedMonth',
        'CalculatedYear': '$CalculatedYear',
        'Nationality': '$Nationality',
        'LodgingRevenue': '$LodgingRevenue',
        'OtherRevenue': '$OtherRevenue'
    }
}, {
    '$group': {
        '_id': None,
        'minDate': {
            '$min': '$CalculatedYear'
        },
        'maxDate': {
            '$max': '$CalculatedYear'
        },
        'data': {
            '$push': '$$ROOT'
        }
    }
}, {
    '$unwind': {
        'path': '$data'
    }
}, {
    '$set': {
        'count': {
            '$subtract': [
                '$maxDate', '$minDate'
            ]
        }
    }
}, {
```

```
        '$project': {
            'CalculatedMonth': '$data.CalculatedMonth',
            'CalculatedYear': '$data.CalculatedYear',
            'Nationality': '$data.Nationality',
            'LodgingRevenue': '$data.LodgingRevenue',
            'OtherRevenue': '$data.OtherRevenue',
            'count': '$count'
        }
    }, {
        '$addFields': {
            'deviationFromRevenue': {
                '$multiply': [
                    {
                        '$rand': {}
                    }, 10
                ]
            }
        }
    }, {
        '$group': {
            '_id': '$CalculatedMonth',
            'LodgingRevenueForMonth': {
                '$sum': '$LodgingRevenue'
            },
            'OtherRevenueForMonth': {
                '$sum': '$OtherRevenue'
            },
            'deviationFromRevenue': {
                '$sum': '$deviationFromRevenue'
            },
            'NumYears': {
                '$push': '$count'
            }
        }
    }, {
        '$addFields': {
            'TotalRevenue': {
                '$subtract': [
                    {
                        '$add': [
                            '$LodgingRevenueForMonth', '$OtherRevenueForMonth'
                        ]
                    }, '$deviationFromRevenue'
                ]
            },
            'NumYears': {
```

```
                '$arrayElemAt': [
                    '$NumYears', 0
                ]
            }
        }
    }, {
        '$sort': {
            '_id': 1
        }
    }, {
        '$project': {
            'Month': {
                '$cond': {
                    'if': {
                        '$lte': [
                            '$_id', 9
                        ]
                    },
                    'then': {
                        '$concat': [
                            '0', {
                                '$toString': '$_id'
                            }
                        ]
                    },
                    'else': {
                        '$toString': '$_id'
                    }
                }
            },
            'TotalCountOfRecordsForMonth': '$NumYears',
            'LodgingRevenueForMonth': '$LodgingRevenueForMonth',
            'OtherRevenueForMonth': '$OtherRevenueForMonth',
            'TotalRevenue': '$TotalRevenue'
        }
    }, {
        '$merge': {
            'into': 'MonthlyTotalRevenue'
        }
    }
]

trainDataMontly = list(collection.aggregate(trainDataMontlyPipeline))

#Test data and predict next 6 month data
predictRevenuePipeline = [
```

```
{
    '$addFields': {
        'CalculatedMonth': {
            '$let': {
                'vars': {
                    'rawMonth': {
                        '$month': {
                            '$subtract': [
                                {
                                    '$toDate': datetime.utcnow()
                                }, {
                                    '$multiply': [
                                        '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                                    ]
                                }
                            ]
                        }
                    }
                },
                'in': {
                    '$cond': {
                        'if': {
                            '$lte': [
                                '$$rawMonth', 9
                            ]
                        },
                        'then': {
                            '$concat': [
                                '0', {
                                    '$toString': '$$rawMonth'
                                }
                            ]
                        },
                        'else': {
                            '$toString': '$$rawMonth'
                        }
                    }
                }
            }
        },
        'CalculatedYear': {
            '$year': {
                '$subtract': [
                    {
                        '$toDate': datetime.utcnow()
                    }, {
```

```
                        '$multiply': [
                            '$DaysSinceLastStay', 24 * 60 * 60 * 1000
                        ]
                    }
                ]
            }
        },
        'calculatedDate': {
            '$subtract': [
                {
                    '$toDate': datetime.utcnow()
                }, {
                    '$multiply': [
                        '$DaysSinceCreation', 24 * 60 * 60 * 1000
                    ]
                }
            ]
        }
    }
}, {
    '$match': {
        'BookingsCheckedIn': {
            '$eq': 1
        }
    }
}, {
    '$group': {
        '_id': {
            'YearGroup': '$CalculatedYear',
            'MonthGroup': '$CalculatedMonth'
        },
        'count': {
            '$sum': 1
        },
        'TotalLodgingRevenue': {
            '$sum': '$LodgingRevenue'
        },
        'TotalOtherRevenue': {
            '$sum': '$OtherRevenue'
        }
    }
}, {
    '$addFields': {
        'TotalRevenue': {
            '$add': [
                '$TotalLodgingRevenue', '$TotalOtherRevenue'
```

```
            ]
        }
    }
}, {
    '$sort': {
        '_id.YearGroup': 1,
        '_id.MonthGroup': 1
    }
}, {
    '$group': {
        '_id': '1',
        'YearAndMonthGroup': {
            '$push': {
                '$concat': [
                    {
                        '$toString': '$_id.MonthGroup'
                    }, '-', {
                        '$toString': '$_id.YearGroup'
                    }
                ]
            }
        },
        'revenue': {
            '$push': '$TotalRevenue'
        }
    }
}, {
    '$addFields': {
        'SMA': {
            '$map': {
                'input': {
                    '$range': [
                        0, {
                            '$size': '$revenue'
                        }
                    ]
                },
                'in': {
                    '$avg': {
                        '$slice': [
                            '$revenue', {
                                '$cond': {
                                    'if': {
                                        '$lt': [
                                            {
                                                '$subtract': [
```

```
                                                {
                                                    '$add': [
                                                        '$$this', 1
                                                    ]
                                                }, 6
                                            ]
                                        }, 0
                                    ]
                                },
                                'then': 0,
                                'else': {
                                    '$subtract': [
                                        {
                                            '$add': [
                                                '$$this', 1
                                            ]
                                        }, 6
                                    ]
                                }
                            }
                        }, 6
                    ]
                }
            }
        }
    }
}, {
    '$addFields': {
        'SMA': {
            '$concatArrays': [
                '$SMA', {
                    '$map': {
                        'input': {
                            '$range': [
                                {
                                    '$subtract': [
                                        {
                                            '$size': '$SMA'
                                        }, 6
                                    ]
                                }, {
                                    '$add': [
                                        {
                                            '$size': '$SMA'
                                        }, 0
```

```
                                ]
                            }
                        ]
                    },
                    'in': {
                        '$avg': {
                            '$slice': [
                                '$revenue', '$$this', 6
                            ]
                        }
                    }
                }
            }
        ]
    }
}, {
    '$set': {
        'lastElement': {
            '$arrayElemAt': [
                '$YearAndMonthGroup', -1
            ]
        }
    }
}, {
    '$set': {
        'nextMonthYear': {
            '$dateToString': {
                'format': '%m-%Y',
                'date': {
                    '$dateFromParts': {
                        'year': {
                            '$toInt': {
                                '$arrayElemAt': [
                                    {
                                        '$split': [
                                            '$lastElement', '-'
                                        ]
                                    }, 1
                                ]
                            }
                        },
                        'month': {
                            '$add': [
                                {
                                    '$toInt': {
```

```
                                        '$arrayElemAt': [
                                            {
                                                '$split': [
                                                    '$lastElement', '-'
                                                ]
                                            }, 0
                                        ]
                                    }
                                }, 1
                            ]
                        },
                        'day': 1
                    }
                }
            }
        }
    }, {
        '$set': {
            'YearAndMonthGroup': {
                '$concatArrays': [
                    '$YearAndMonthGroup', [
                        '$nextMonthYear'
                    ]
                ]
            }
        }
    }, {
        '$set': {
            'lastElement': {
                '$arrayElemAt': [
                    '$YearAndMonthGroup', -1
                ]
            }
        }
    }, {
        '$set': {
            'nextMonthYear': {
                '$dateToString': {
                    'format': '%m-%Y',
                    'date': {
                        '$dateFromParts': {
                            'year': {
                                '$toInt': {
                                    '$arrayElemAt': [
                                        {
```

```
                                        '$split': [
                                            '$lastElement', '-'
                                        ]
                                    }, 1
                                ]
                            }
                        },
                        'month': {
                            '$add': [
                                {
                                    '$toInt': {
                                        '$arrayElemAt': [
                                            {
                                                '$split': [
                                                    '$lastElement', '-'
                                                ]
                                            }, 0
                                        ]
                                    }
                                }, 1
                            ]
                        },
                        'day': 1
                    }
                }
            }
        }
    }, {
        '$set': {
            'YearAndMonthGroup': {
                '$concatArrays': [
                    '$YearAndMonthGroup', [
                        '$nextMonthYear'
                    ]
                ]
            }
        }
    }, {
        '$set': {
            'lastElement': {
                '$arrayElemAt': [
                    '$YearAndMonthGroup', -1
                ]
            }
        }
```

```
    }, {
        '$set': {
            'nextMonthYear': {
                '$dateToString': {
                    'format': '%m-%Y',
                    'date': {
                        '$dateFromParts': {
                            'year': {
                                '$toInt': {
                                    '$arrayElemAt': [
                                        {
                                            '$split': [
                                                '$lastElement', '-'
                                            ]
                                        }, 1
                                    ]
                                }
                            },
                            'month': {
                                '$add': [
                                    {
                                        '$toInt': {
                                            '$arrayElemAt': [
                                                {
                                                    '$split': [
                                                        '$lastElement', '-'
                                                    ]
                                                }, 0
                                            ]
                                        }
                                    }, 1
                                ]
                            },
                            'day': 1
                        }
                    }
                }
            }
        }
    }, {
        '$set': {
            'YearAndMonthGroup': {
                '$concatArrays': [
                    '$YearAndMonthGroup', [
                        '$nextMonthYear'
                    ]
```

```
            ]
        }
    }
}, {
    '$set': {
        'lastElement': {
            '$arrayElemAt': [
                '$YearAndMonthGroup', -1
            ]
        }
    }
}, {
    '$set': {
        'nextMonthYear': {
            '$dateToString': {
                'format': '%m-%Y',
                'date': {
                    '$dateFromParts': {
                        'year': {
                            '$toInt': {
                                '$arrayElemAt': [
                                    {
                                        '$split': [
                                            '$lastElement', '-'
                                        ]
                                    }, 1
                                ]
                            }
                        },
                        'month': {
                            '$add': [
                                {
                                    '$toInt': {
                                        '$arrayElemAt': [
                                            {
                                                '$split': [
                                                    '$lastElement', '-'
                                                ]
                                            }, 0
                                        ]
                                    }
                                }, 1
                            ]
                        },
                        'day': 1
                    }
```

```
                }
            }
        }
    }
}, {
    '$set': {
        'YearAndMonthGroup': {
            '$concatArrays': [
                '$YearAndMonthGroup', [
                    '$nextMonthYear'
                ]
            ]
        }
    }
}, {
    '$set': {
        'lastElement': {
            '$arrayElemAt': [
                '$YearAndMonthGroup', -1
            ]
        }
    }
}, {
    '$set': {
        'nextMonthYear': {
            '$dateToString': {
                'format': '%m-%Y',
                'date': {
                    '$dateFromParts': {
                        'year': {
                            '$toInt': {
                                '$arrayElemAt': [
                                    {
                                        '$split': [
                                            '$lastElement', '-'
                                        ]
                                    }, 1
                                ]
                            }
                        },
                        'month': {
                            '$add': [
                                {
                                    '$toInt': {
                                        '$arrayElemAt': [
                                            {
```

```
                                                '$split': [
                                                    '$lastElement', '-'
                                                ]
                                            }, 0
                                        ]
                                    }
                                }, 1
                            ]
                        },
                        'day': 1
                    }
                }
            }
        }
    }, {
        '$set': {
            'YearAndMonthGroup': {
                '$concatArrays': [
                    '$YearAndMonthGroup', [
                        '$nextMonthYear'
                    ]
                ]
            }
        }
    }, {
        '$set': {
            'lastElement': {
                '$arrayElemAt': [
                    '$YearAndMonthGroup', -1
                ]
            }
        }
    }, {
        '$set': {
            'nextMonthYear': {
                '$dateToString': {
                    'format': '%m-%Y',
                    'date': {
                        '$dateFromParts': {
                            'year': {
                                '$toInt': {
                                    '$arrayElemAt': [
                                        {
                                            '$split': [
                                                '$lastElement', '-'
```

```
                                    ]
                                }, 1
                            ]
                        }
                    },
                    'month': {
                        '$add': [
                            {
                                '$toInt': {
                                    '$arrayElemAt': [
                                        {
                                            '$split': [
                                                '$lastElement', '-'
                                            ]
                                        }, 0
                                    ]
                                }
                            }, 1
                        ]
                    },
                    'day': 1
                }
            }
        }
    }
}, {
    '$set': {
        'YearAndMonthGroup': {
            '$concatArrays': [
                '$YearAndMonthGroup', [
                    '$nextMonthYear'
                ]
            ]
        }
    }
}, {
    '$set': {
        'revenue': {
            '$concatArrays': [
                '$revenue', [
                    0, 0, 0, 0, 0, 0
                ]
            ]
        }
    }
```

```
    }, {
        '$project': {
            '_id': 0,
            'zip': {
                '$zip': {
                    'inputs': [
                        '$YearAndMonthGroup', '$revenue', '$SMA'
                    ]
                }
            }
        }
    }, {
        '$unwind': {
            'path': '$zip',
            'preserveNullAndEmptyArrays': True
        }
    }, {
        '$project': {
            'Month': {
                '$arrayElemAt': [
                    {
                        '$split': [
                            {
                                '$arrayElemAt': [
                                    '$zip', 0
                                ]
                            }, '-'
                        ]
                    }, 0
                ]
            },
            'MonthYear': {
                '$arrayElemAt': [
                    '$zip', 0
                ]
            },
            'ActualRevenue': {
                '$arrayElemAt': [
                    '$zip', 1
                ]
            },
            'ProjectedRevenue': {
                '$arrayElemAt': [
                    '$zip', 2
                ]
            }
```

```
        }
    }, {
        '$lookup': {
            'from': 'MonthlyTotalRevenue',
            'localField': 'Month',
            'foreignField': 'Month',
            'as': 'result'
        }
    }, {
        '$unwind': {
            'path': '$result'
        }
    }, {
        '$addFields': {
            'ProjectedRevenue': {
                '$divide': [
                    {
                        '$add': [
                            '$ProjectedRevenue', '$result.TotalRevenue'
                        ]
                    }, {
                        '$add': [
                            '$result.TotalCountOfRecordsForMonth', 1
                        ]
                    }
                ]
            }
        }
    }, {
        '$project': {
            'Month': {
                '$arrayElemAt': [
                    {
                        '$split': [
                            '$MonthYear', '-'
                        ]
                    }, 0
                ]
            },
            'MonthYear': '$MonthYear',
            'ActualRevenue': '$ActualRevenue',
            'ProjectedRevenue': '$ProjectedRevenue'
        }
    }
]
```

```
predictedActualData = list(collection.aggregate(predictRevenuePipeline))

predictedActualRevenueData = [entry for entry in predictedActualData if
entry['ActualRevenue'] > 0]  # You can adjust the condition as needed

month_years = [entry['MonthYear'] for entry in predictedActualData]
actual_revenue = [entry['ActualRevenue'] for entry in predictedActualRevenueData]
projected_revenue = [entry['ProjectedRevenue'] for entry in predictedActualData]

fig = go.Figure()

# Add traces for actual and projected revenue
fig.add_trace(go.Scatter(x=month_years, y=actual_revenue, mode='lines+markers',
name='Actual Revenue'))
fig.add_trace(go.Scatter(x=month_years, y=projected_revenue, mode='lines+markers',
name='Projected Revenue'))

# Customize the layout
fig.update_layout(
    title='Actual vs Projected Revenue Over Months',
    xaxis_title='Month-Year',
    yaxis_title='Revenue',
    xaxis=dict(tickangle=45),
    hovermode='closest',
    height=750,
    width=1000
)

st.plotly_chart(fig)



# print(topSpendingNationality)

df = pd.DataFrame([(month, nat, spend) for entry in topSpendingNationality for month,
nats, spends in [(entry['month'], entry['topNationalities'], entry['topSpending'])] for
nat, spend in zip(nats, spends)], columns=['Month', 'Nationality', 'Spending'])

# Create an interactive bar chart
fig = px.bar(df, x='Nationality', y='Spending', color='Month', title='Range of Spending by
Nationality Each Month',
             labels={'Spending': 'Spending Amount', 'Month': 'Month Number'},
barmode='group')

# Show the interactive chart using Streamlit
# st.plotly_chart(fig)
```

```
########################################
# months = [entry['month'] for entry in topSpendingNationality]
# top_nationalities = [entry['topNationalities'][0] for entry in topSpendingNationality]
# top_spending = [entry['topSpending'][0] for entry in topSpendingNationality]

# fig = px.scatter(topSpendingNationality, x=months, y=top_spending,
color=top_nationalities, labels={'x': 'Month', 'y': 'Top Spending'},
#                  title='Top Nationalities and Spending by Month')
# st.plotly_chart(fig)

months = [entry['month'] for entry in topSpendingNationality]
top_nationalities_0 = [entry['topNationalities'][0] for entry in topSpendingNationality]
top_spending_0 = [entry['topSpending'][0] for entry in topSpendingNationality]
top_nationalities_1 = [entry['topNationalities'][1] for entry in topSpendingNationality]
top_spending_1 = [entry['topSpending'][1] for entry in topSpendingNationality]

fig = go.Figure()

# Add a trace for each nationality in the first entry
for nationality in set(top_nationalities_0):
    indices = [i for i, value in enumerate(top_nationalities_0) if value == nationality]
    fig.add_trace(go.Scatter(x=[months[i] for i in indices],
                             y=[top_spending_0[i] for i in indices],
                             mode='markers',
                             name=f'{nationality}'))

# Add a trace for each nationality in the second entry
for nationality in set(top_nationalities_1):
    indices = [i for i, value in enumerate(top_nationalities_1) if value == nationality]
    print([months[i] for i in indices])
    fig.add_trace(go.Scatter(x=[months[i] for i in indices],
                             y=[top_spending_1[i] for i in indices],
                             mode='markers',
                             name=f'{nationality}'))

fig.update_layout(title='Top Nationalities and Spending by Month',
                  xaxis=dict(title='Month', tickmode='array', tickvals=months),
                  yaxis=dict(title='Top Spending'))

# st.plotly_chart(fig)
```

Code for Customer Review Analysis

```
from pymongo import MongoClient
from wordcloud import WordCloud
import matplotlib.pyplot as plt
import streamlit as st
from datetime import datetime, tzinfo, timezone
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go

# Page configurations
PAGE_CONFIG = {"page_title": "Word Analysis", "page_icon": ":brain"}

# Set page title and icon
st.set_page_config(layout="wide", page_title=PAGE_CONFIG["page_title"],
page_icon=PAGE_CONFIG["page_icon"])

# Sidebar menu
# menu_options = ["Page 1", "Page 2", "Page 3"]
# selected_page = st.sidebar.radio("Select a Page", menu_options)

# Connect to MongoDB
client = MongoClient('mongodb://localhost:27017/')
db = client['cs532']['HotelWithReviews']

# Your MongoDB aggregation query
pipeline = [
    {
        "$project": {
            "words": {"$split": [{'$toString':"$CustomerReview"}, " "]}
        }
    },
    {
        "$unwind": "$words"
    },
    {
        "$group": {
            "_id": "$words",
            "count": {"$sum": 1}
        }
    },
    {
        "$sort": {"count": -1}
    },
    {
        "$limit": 5000
```

```
    }
]

result = list(db.aggregate(pipeline))

# Extract words and counts from the result
words = [item['_id'] for item in result if item['_id'] not in ['a', 'an', 'the', 'in',
'on', 'at', 'for', 'to', 'with', 'by',
'of','for','I','was','and','The','were','is','that','you','have','but','so','it','as','we'
,'very','room','not','staff','No','from','My','t','hotel','location','had','Location','bre
kfast','are','be','small','my','Negative']]
counts = [item['count'] for item in result]

# Create a dictionary with words and their counts
word_count_dict = dict(zip(words, counts))

wordcloud = WordCloud(width=800, height=400,
background_color='white').generate_from_frequencies(word_count_dict)



pipeline = [
    {
        "$project": {
            "words": {"$split": [{'$toString':"$CustomerReview"}, " "]}
        }
    },
    {
        "$unwind": "$words"
    },
    {
        "$group": {
            "_id": "$words",
            "count": {"$sum": 1}
        }
    },
    {
        "$sort": {"count": -1}
    },
    {
        "$limit": 100
    }
]

result = list(db.aggregate(pipeline))
result = [doc for doc in result if doc['_id'] and doc['_id'].strip()]
```

```
exclude_list = ['Negative', 'no', 's', 'I', 'No', 'i', 'a', 'an', 'the', 'in', 'on', 'at',
'for', 'to', 'with', 'by', 'of', 'for', 'I', 'was', 'and', 'The', 'were', 'is', 'that',
'you', 'have', 'but', 'so', 'it', 'as', 'we', 'very', 'room', 'not', 'staff', 'No',
'from', 'My', 't', 'hotel', 'location', 'had', 'Location', 'brekfast', 'are', 'be',
'small', 'my', ]

# Remove documents with empty or null '_id' field and exclude specific words
result = [doc for doc in result if doc['_id'] and doc['_id'].strip() and
doc['_id'].lower() not in exclude_list]


st.title("Word Frequency Analysis")

# Plot the WordCloud image using Streamlit
st.image(wordcloud.to_image(),use_column_width=True)

# Plot a bar chart using Plotly
fig = px.bar(result, x='_id', y='count', title='Word Frequency Bar Chart', labels={'_id':
'Word', 'count': 'Frequency'})
st.plotly_chart(fig, use_container_width=True)


#-------------------------------------------------------------------------------------
-----

word_count_pipeline = [
    {
        '$match': {
            '$expr': {
                '$and': [
                    { '$gte': [{ '$toDouble': "$Age" }, 20] },
                    { '$lte': [{ '$toDouble': "$Age" }, 40] }
                ]
            }
        }
    }, {
        '$project': {
            'CustomerReview': {
                '$toString': '$CustomerReview'
            }
        }
    }, {
        '$project': {
            'top_words': {
                '$split': [
                    '$CustomerReview', ' '
```

```
                ]
            }
        }
    }, {
        '$unwind': {
            'path': '$top_words',
            'preserveNullAndEmptyArrays': False
        }
    }, {
        '$group': {
            '_id': '$top_words',
            'count': {
                '$sum': 1
            }
        }
    }, {
        '$sort': {
            'count': -1
        }
    }, {
        '$limit': 400
    }, {
        '$addFields': {
            'word_classification': {
                '$switch': {
                    'branches': [
                        {
                            'case': {
                                '$in': [
                                    '$_id', [
                                        "helpful", "friendly", "clean", "nice",
"comfortable", "great", "good", "excellent", "positive", "lovely", "perfect", "comfy",
"quiet", "amazing", "nice", "spacious", "beautiful", "right", "welcoming", "recommend",
"wonderful", "convenient", "polite", "liked", "fine", "attentive", "top"
                                    ]
                                ]
                            },
                            'then': 'positive'
                        }, {
                            'case': {
                                '$in': [
                                    '$_id', [
                                        "not", "negative", "expensive", "noisy", "poor",
"didn't", "noise", "bad", "slow", "problem"
                                    ]
                                ]
```

```
                            },
                            'then': 'negative'
                        }
                    ],
                    'default': 'neutral'
                }
            }
        }
    }, {
        '$group': {
            '_id': '$word_classification',
            'word_classifications': {
                '$addToSet': '$_id'
            }
        }
    }, {
        '$project': {
            '_id': 1,
            'word_classifications': {
                '$reduce': {
                    'input': '$word_classifications',
                    'initialValue': '',
                    'in': {
                        '$concat': [
                            '$$value', '|', '$$this'
                        ]
                    }
                }
            }
        }
    }
]

wc_result = list(db.aggregate(word_count_pipeline))
pos = [entry['word_classifications'] for entry in wc_result if entry['_id'] ==
'positive'][0]
neg = [entry['word_classifications'] for entry in wc_result if entry['_id'] ==
'negative'][0]
pos = pos.strip('|')
neg = neg.strip('|')
# print(pos)


sentiment_pipeline = [
    {
        '$match': {
```

```
            '$expr': {
                '$and': [
                    {'$gte': [{'$toDouble': '$Age'}, 20]},
                    {'$lte': [{'$toDouble': '$Age'}, 40]}
                ]
            }
        }
    },
    {
        '$project': {
            '_id': 1,
            'Nationality': '$Nationality',
            'Age': '$Age',
            'SRHighFloor': '$SRHighFloor',
            'SRLowFloor': '$SRLowFloor',
            'SRAccessibleRoom': '$SRAccessibleRoom',
            'SRMediumFloor': '$SRMediumFloor',
            'SRBathtub': '$SRBathtub',
            'SRShower': '$SRShower',
            'CustomerReview': {'$toString':'$CustomerReview'}
        }
    },
    {
        '$addFields': {
            'sentiment': {
                '$cond': {
                    'if': {
                        '$regexMatch': {
                            'input': '$CustomerReview',
                            'regex': pos,
                            'options': 'i'
                        }
                    },
                    'then': 'positive',
                    'else': {
                        '$cond': {
                            'if': {
                                '$regexMatch': {
                                    'input': '$CustomerReview',
                                    'regex': neg,
                                    'options': 'i'
                                }
                            },
                            'then': 'negative',
                            'else': 'neutral'
                        }
```

```
                    }
                }
            }
        }
    },
    {
        '$project': {
            '_id': 1,
            'Nationality': '$Nationality',
            'Age': '$Age',
            'SRHighFloor': '$SRHighFloor',
            'SRLowFloor': '$SRLowFloor',
            'SRAccessibleRoom': '$SRAccessibleRoom',
            'SRMediumFloor': '$SRMediumFloor',
            'SRBathtub': '$SRBathtub',
            'SRShower': '$SRShower',
            'CustomerReview': '$CustomerReview',
            'Sentiment': '$sentiment'
        }
    },
    {
        '$group': {
            '_id': '$Nationality',
            'pos_count': {'$sum': {'$cond': [{'$eq': ['$Sentiment', 'positive']}, 1, 0]}},
            'neg_count': {'$sum': {'$cond': [{'$eq': ['$Sentiment', 'negative']}, 1, 0]}},
            'neut_count': {'$sum': {'$cond': [{'$eq': ['$Sentiment', 'neutral']}, 1, 0]}}
        }
    }
]

sentiment_result = list(db.aggregate(sentiment_pipeline))

nationalities = [entry['_id'] for entry in sentiment_result]
pos_count = [entry['pos_count'] for entry in sentiment_result]
neg_count = [entry['neg_count'] for entry in sentiment_result]
neut_count = [entry['neut_count'] for entry in sentiment_result]


fig_sentiment = go.Figure()

fig_sentiment.add_trace(go.Bar(x=nationalities, y=pos_count, name='Positive',
marker_color='green'))
fig_sentiment.add_trace(go.Bar(x=nationalities, y=neut_count, name='Neutral',
marker_color='blue'))
fig_sentiment.add_trace(go.Bar(x=nationalities, y=neg_count, name='Negative',
marker_color='red'))
```

```
# Update the layout
fig_sentiment.update_layout(
    title='Geospatial Sentiment Analysis',
    xaxis_title='Nationality',
    yaxis_title='Count',
    barmode='group'  # Grouped bar chart
)

st.title("Geospatial Sentiment Analysis")

st.plotly_chart(fig_sentiment, use_container_width=True)
```