

Computer Engineering Training Program for Engineers

BI-ML Project Report

Prediction of Fake Reservations for British Airways

Made By

TORKHANI Oumayma

KHALED Ghassen

BEN HMIDA Mohammad Ali

TOKOGMON Djavph David

Academic Year : 2022/2023



Table of Contents

Contents

Introduction.....	6
Chapter 1: Introduction	7
1.1 Background	7
1.2 Objectives.....	7
1.3 Problem Statement.....	7
1.4 Web Scraping and Dataset.....	7
1.5 Scrum Methodology	8
1.6 GitHub	12
1.7 CRISP-DM for Machine Learning	14
1.8 Conclusion.....	15
Chapter 2: Business Intelligence (BI)	16
2.1 Definition and Quick Differentiation of BI	16
2.2 Data Source	16
2.3 Data Preparation	17
2.4 Creating Database Objects	17
2.4.1 ETL Source to Staging	18
2.4.2 ETL Staging to DW	18
2.5 Semantic Layer	20
2.5.1 SSIS Tabular and Multidimensional	20
2.5.1 SSMS Cube	22
2.6 Creating Power BI Report.....	22
2.7 Conclusion.....	24
Chapter 3: Machine learning (ML)	25
3.1 Definition and Quick Differentiation of ML	25

3.2 Steps for Developing a Machine Learning Model	25
3.2.1 Business understanding	25
3.2.2 Data understanding	25
3.2.3 Data preparation	26
3.2.4 Modeling	30
3.2.5 Evaluation	31
3.2.6 Deployment	33
3.3 Conclusion.....	33
Chapter 4: Deployment.....	34
4.1 Definition and Quick Differentiation of ML Deployment	34
4.2 Django	34
4.3 Web Application with Django.....	34
4.5 ML Model - Deployment	36
4.6 Conclusion.....	40
Conclusion.....	41

List of Figures

Figure 1 Screenshot of Trello page with name of members of group	8
Figure 2 Screenshot of different states.....	9
Figure 3 Screenshot of Sprints of the project (Report, BI Project, ML Project, deployment)	9
Figure 4 Screenshot of different tasks in Report Sprint.	10
Figure 5 Screenshot of different tasks in BI Sprint.....	10
Figure 6 Screenshot of different tasks in ML Sprint.	11
Figure 7 Screenshot of different tasks in deployment Sprint.....	11
Figure 8 Screenshot of the first step of sending request to collaborate project code.....	12
Figure 9 Screenshot of central repository "Fake-reservation-check "	13
Figure 10 Screenshot of BI Project repository named "Fake-reservation-check-BI"	13
Figure 11 Screenshot of web and deployment Project repository named "Fake-reservation-check-WEB"	14
Figure 12 Screenshot of ML Project repository named "Fake-reservation-check-ML"	14
Figure 13 Screenshot of part of Web Scraping Code	17
Figure 14 Screenshot of newone database	17
Figure 15 Graph explained ETL Source to Staging	18
Figure 16 Screenshot of the flow of charging from csv file to staging table	18
Figure 17 Graph explained ETL staging to Datawarehouse	18
Figure 18 Screenshot of the flow of charging from staging table to Data warehouse 1	19
Figure 19 Screenshot of the flow of charging from staging table to Data warehouse 2	19
Figure 20 Screenshot of the data warehouse	20
Figure 21 Screenshot of Tabular project using SSIS	21
Figure 22 Screenshot of Cube using SSMS	22
Figure 23 Screenshot of Power BI report 1	23
Figure 24 Screenshot of Power BI report 2.....	23
Figure 25 Screenshot of Power BI report 3.....	24
Figure 26 Screenshot of dataset details	26
Figure 27 Code of Encoder function	27
Figure 28 Code of drop categorical columns	27
Figure 29 Code of Creating Label dataframe.....	27
Figure 30 Code of deleting Label from features dataframe.....	27
Figure 31 Code of Normalizing the values.....	28

Figure 32 Code to create the correlation matrix.....	28
Figure 33 figure of correlation matrix	29
Figure 34 Code of Balancing the dataset.....	29
Figure 35 Code and graph representation of the most important features	30
Figure 36 Code of splitting data into test and train data	30
Figure 37 Code of Creation random forest model and the training	31
Figure 38 Screenshot of labels values.....	31
Figure 39 Code of new model with new balanced dataset	31
Figure 40 Code of evaluation score functions.....	32
Figure 41 Code of testing and evaluating the model	32
Figure 42 Code of new model test.....	32
Figure 43 Screenshot of Project Architect	34
Figure 44 Screenshot of Login Page	35
Figure 45 Screenshot of Dashboard Admin Page	35
Figure 46 Screenshot of Dashboard simple user Page	36
Figure 47 Code using pickle module to creation our ml_model.plk.....	36
Figure 48 Code using pickle module to creation our scaler.plk.....	36
Figure 49 Screenshop of the new files created.....	37
Figure 50 Screenshot of the Model folder architect	37
Figure 51 Code of model predict function	37
Figure 52 Code of view function predict_reservation code	38
Figure 53 Screenshot of a list of reservations awaiting the application of the model.....	38
Figure 54 Screenshot of a list of reservations after the application of the model.....	39
Figure 55 List of prediction with export csv button.....	39
Figure 56 List of prediction in csv file.....	40
Figure 57 Screenshot of a list of reservations after reset the states	40

List of Tables

Table 1 Table of meaning of columns.....	25
Table 2 Divide the columns into features and labels	26

Introduction

In the airline industry, fake reservations can cause a lot of problems for both airlines and passengers. To prevent this, we worked on a project to use Business Intelligence techniques to analyze the data of British Airways' reservation system and predict the likelihood of a reservation being fraudulent. By using machine learning and data analysis tools, we were able to collect and analyze data from multiple sources to identify patterns and predict fraudulent behavior. In this project, we focused on the steps of data collection, cleaning, modeling, analysis, and visualization. The insights we gained from this project can help British Airways and other airlines to detect and eliminate fraudulent activities, which will improve the customer experience and increase the trust of their customers in the airline industry.

This report is divided into four parts: Introduction, Business Intelligence, Machine Learning, and Conclusion.

- In the Introduction section, we provide background information on the project and discuss the definitions of machine learning and Business Intelligence.
- The Business Intelligence section focuses on data collection, cleaning, and analysis.
- The Machine Learning section covers the development and evaluation of the machine learning model.
- In the end we will get a Conclusion that summarizes the key findings and provides recommendations for detecting and eliminating fraudulent activities in the airline industry.

Chapter 1: Introduction

1.1 Background

In the airline industry, fake reservations can cause a lot of problems for both airlines and passengers. Fraudulent activities such as fake bookings, invalid credit cards, and identity theft can lead to significant losses for the airlines and inconvenience for the passengers. Therefore, it is crucial for airlines to detect and eliminate fraudulent activities to improve the customer experience and increase customer trust.

1.2 Objectives

The main objective of this project is to develop a machine learning model to predict the likelihood of a reservation being fraudulent for British Airways. Specifically, the project aims to:

- Collect and preprocess data from multiple sources
- Develop a machine learning model to classify reservations as fraudulent or legitimate
- Evaluate the performance of the model using appropriate metrics
- Provide insights and recommendations based on the results

1.3 Problem Statement

The problem addressed in this project is the need for British Airways to detect and prevent fraudulent activities in their reservation system. The current methods of manual checking and random audits are time-consuming and not efficient enough to detect all fraudulent activities. Therefore, the airline needs an automated system that can analyze data and identify potential fraudulent activities with high accuracy and efficiency.

1.4 Web Scraping and Dataset

To collect data for this project, we will use web scraping to extract information from online booking platforms. One example of a website that we can use to obtain our dataset is Expedia.com. We will scrape data such as flight details, passenger information, and booking history. We will also collect additional data from internal sources within British Airways. The collected data will be preprocessed and used to train and evaluate the machine learning model.

1.5 Scrum Methodology

Scrum is an agile project management framework that emphasizes collaboration, flexibility, and continuous improvement. In Scrum, the project is divided into short iterations called sprints, which typically last between two to four weeks. During each sprint, the team works on a set of tasks and delivers a potentially shippable product increment. Scrum also emphasizes regular team meetings, including daily stand-up meetings, sprint planning meetings, sprint review meetings, and sprint retrospective meetings. The Scrum framework provides a flexible and iterative approach to project management, which can help teams respond quickly to changing requirements and deliver high-quality products.

In our case we use Trello to manage Tasks.

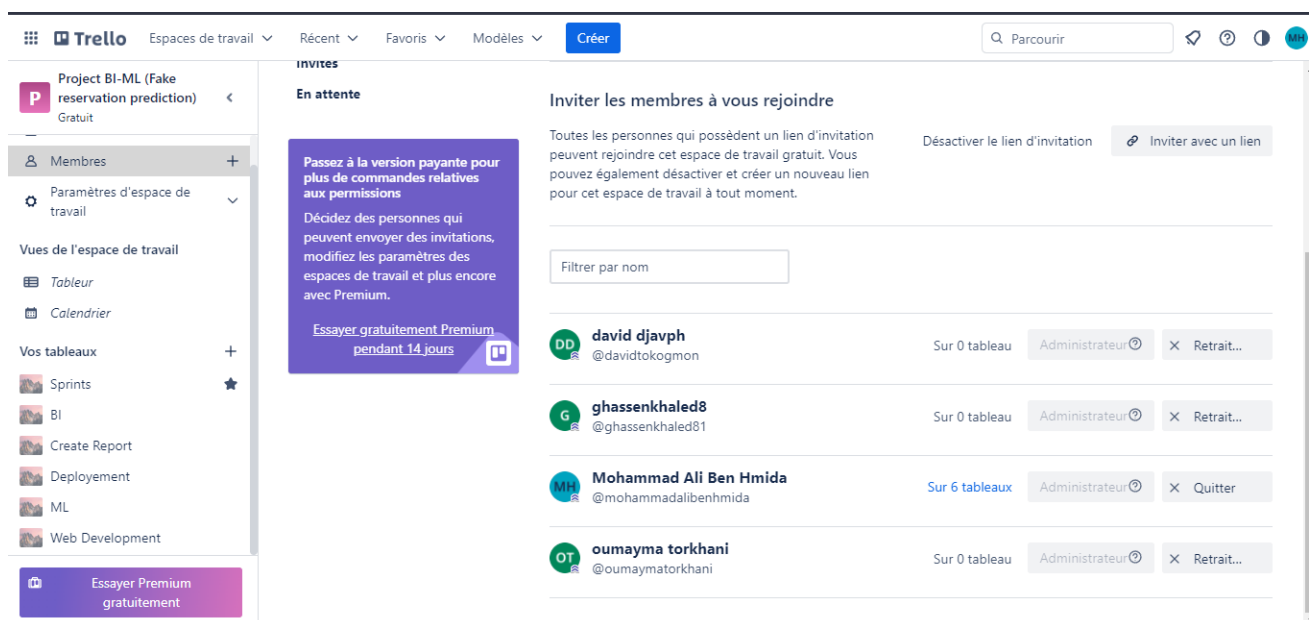


Figure 1 Screenshot of Trello page with name of members of group

For each sprint or task, there are 5 possible states. During our first meeting, we discussed:

To-do: Tasks that need to be completed

In Progress: Tasks that are currently being worked on

Done: Tasks that have been completed

Questions: Tasks that require clarification or further information

To Verify: Tasks that are completed but need to be checked for accuracy or quality.



Figure 2 Screenshot of different states

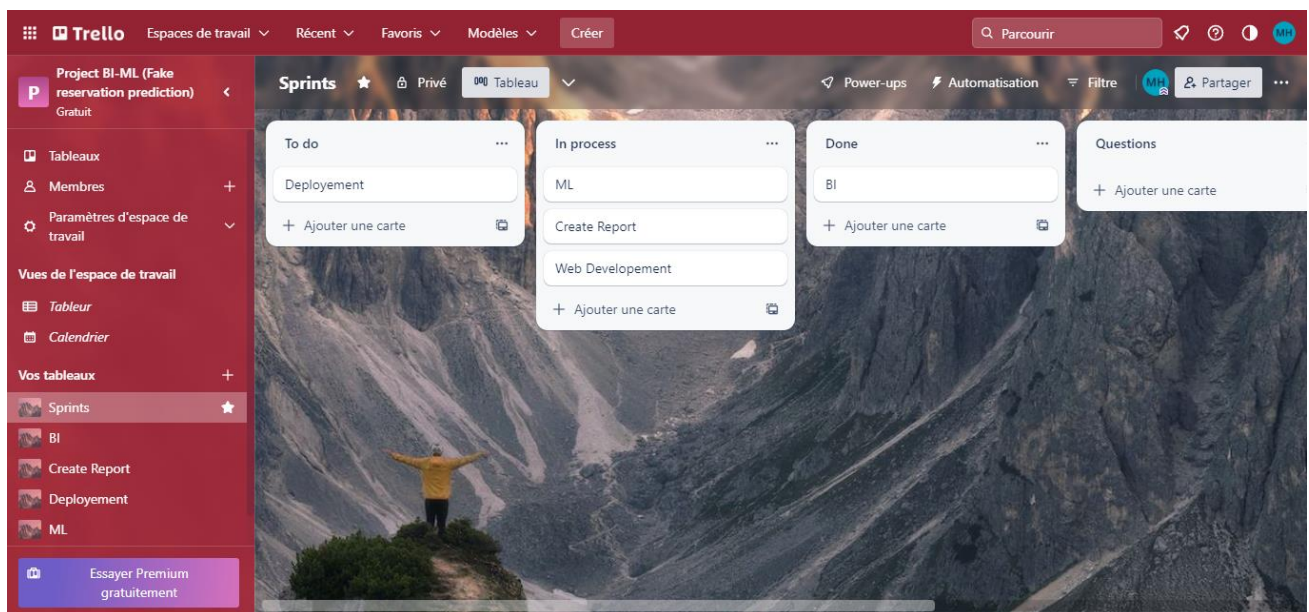


Figure 3 Screenshot of Sprints of the project (Report, BI Project, ML Project, deployment)

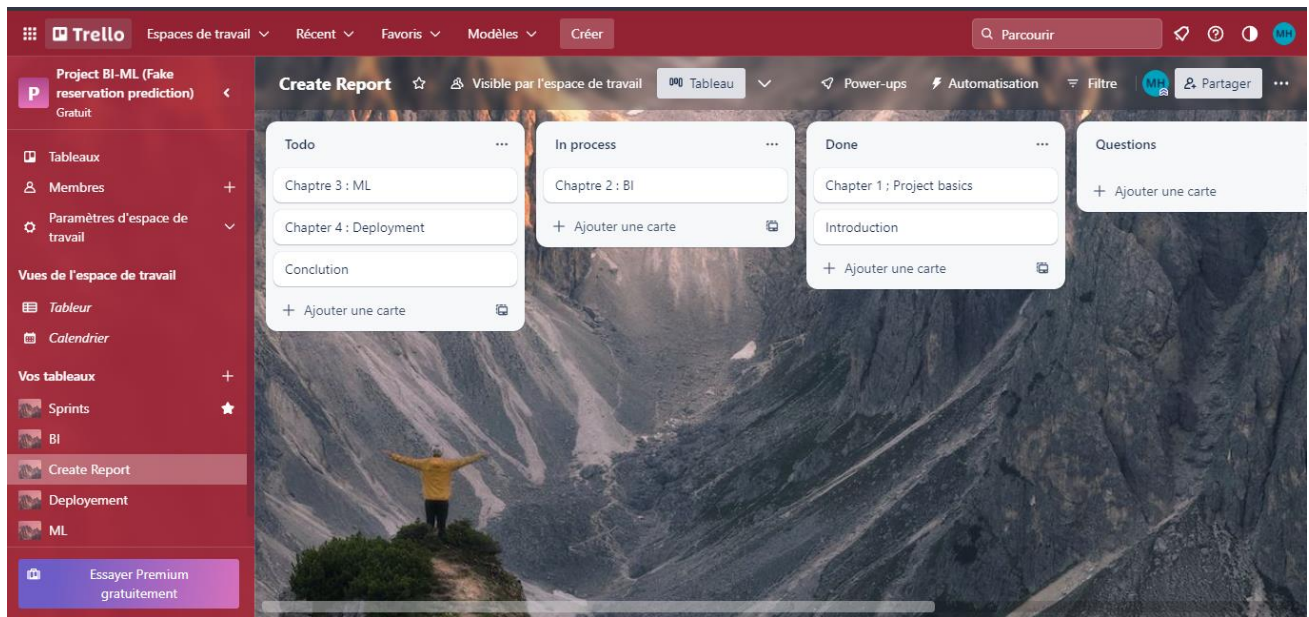


Figure 4 Screenshot of different tasks in Report Sprint.

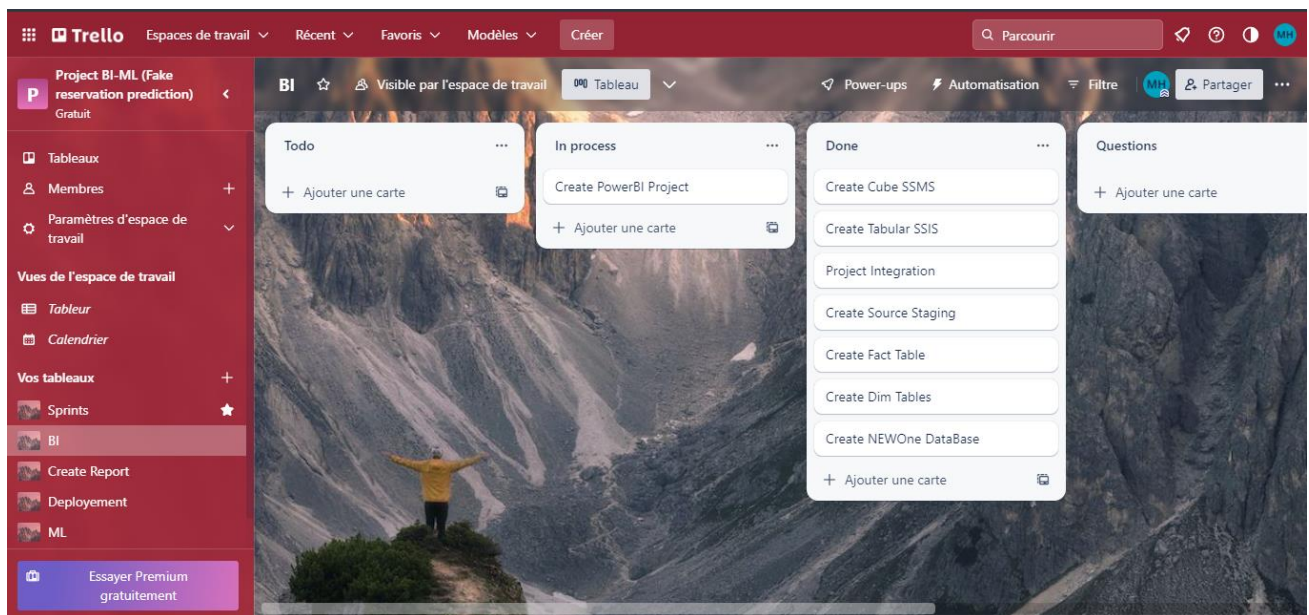


Figure 5 Screenshot of different tasks in BI Sprint.

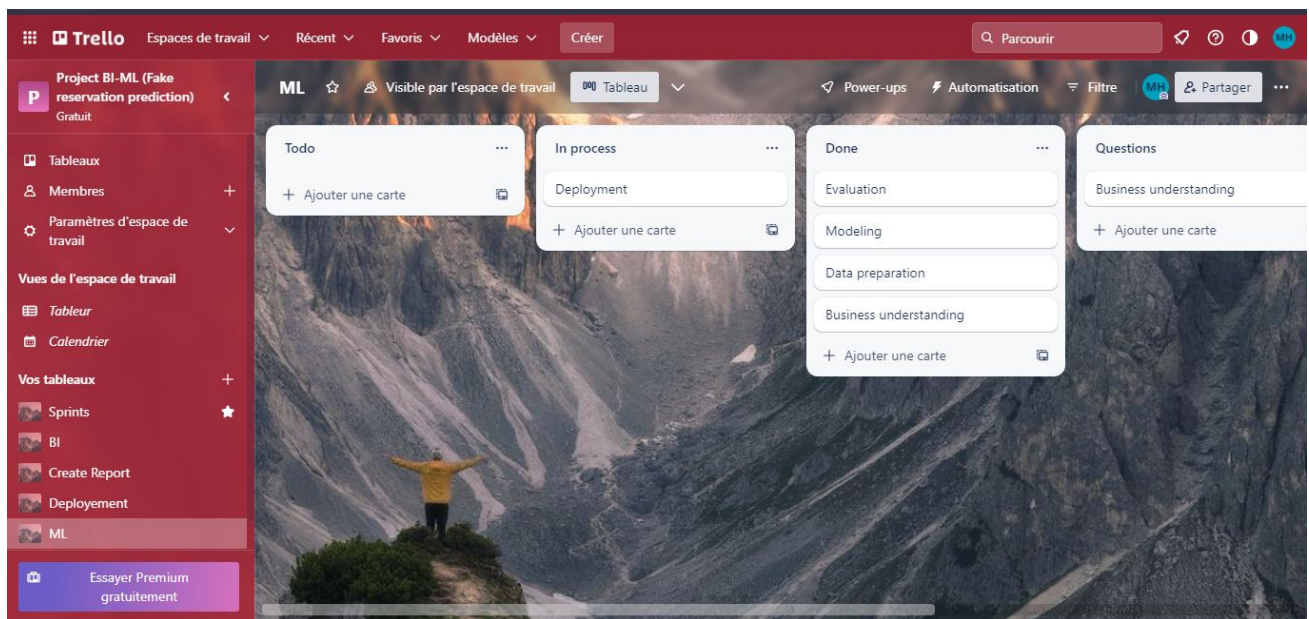


Figure 6 Screenshot of different tasks in ML Sprint.

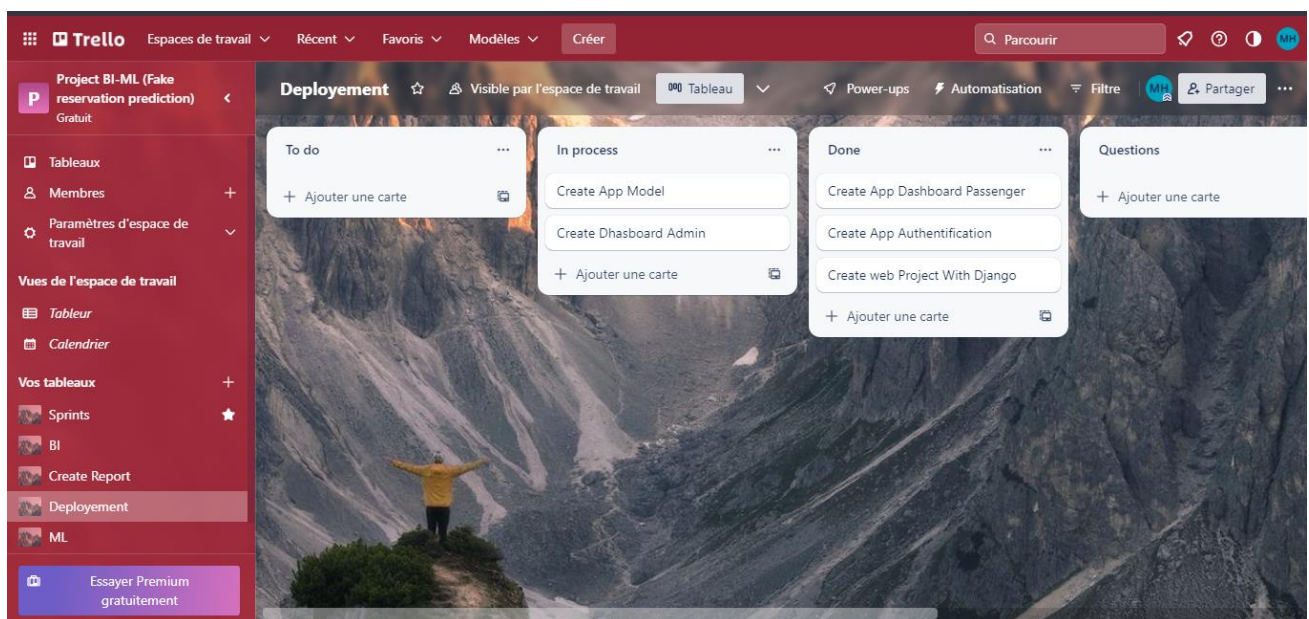


Figure 7 Screenshot of different tasks in deployment Sprint.

1.6 GitHub

GitHub is a web-based platform used for version control and collaboration. It allows developers to store and manage their code repositories in a centralized location, making it easy to track changes, collaborate with others.

In our case our repository in GitHub is private and shared between only our group to the best way to manage different parts of project.

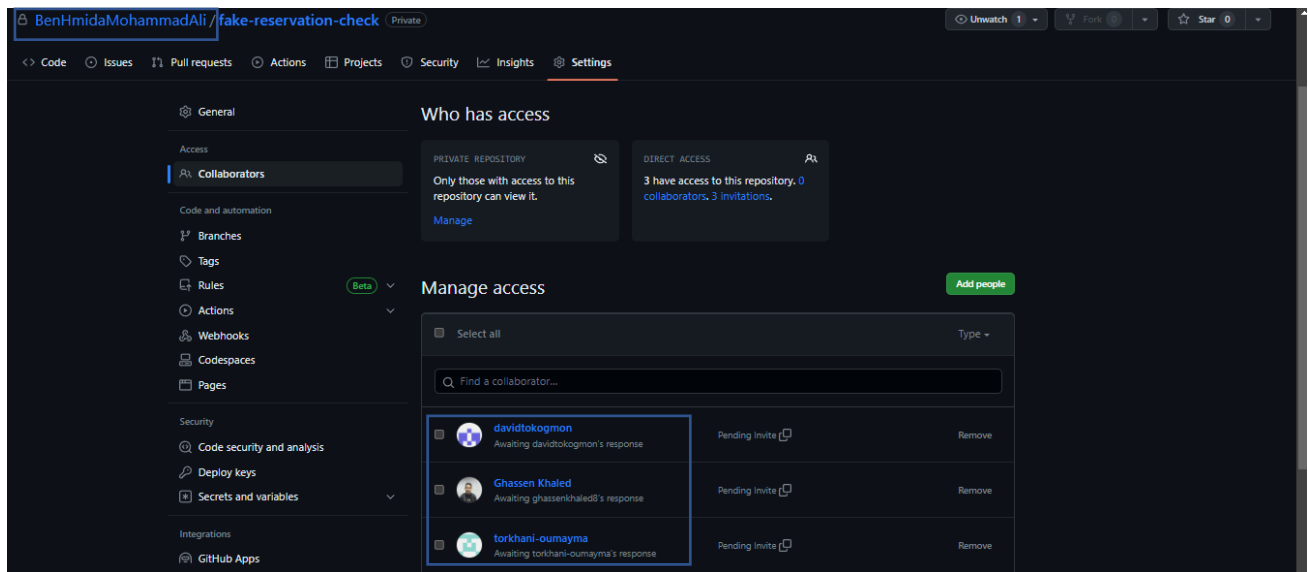


Figure 8 Screenshot of the first step of sending request to collaborate project code

In our specific case, we were working on four different projects: Report, BI, ML, Web, and a deployment project for each one. To streamline our workflow and improve collaboration, we came up with the idea of creating a project called "Fake-reservation-check". This project acts as a central hub, linking all of the other projects together. It makes it easier for our team to manage and keep track of all the code related to the different projects.

(See **figure 9**, **figure 10**, **figure11** and **figure12**)

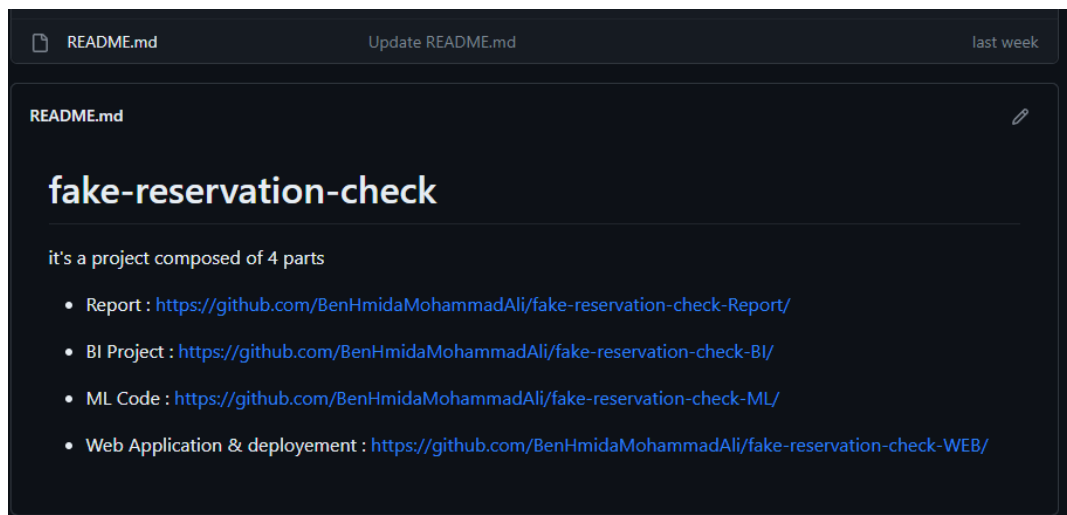


Figure 9 Screenshot of central repository “Fake-reservation-check “

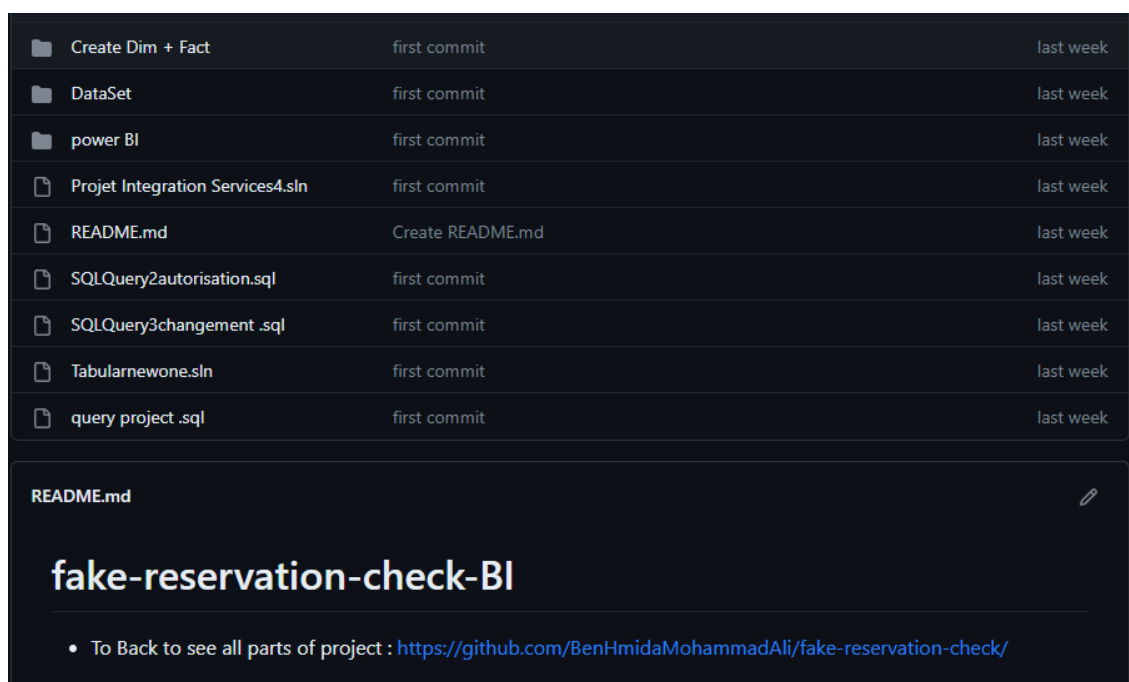


Figure 10 Screenshot of BI Project repository named “Fake-reservation-check-BI“

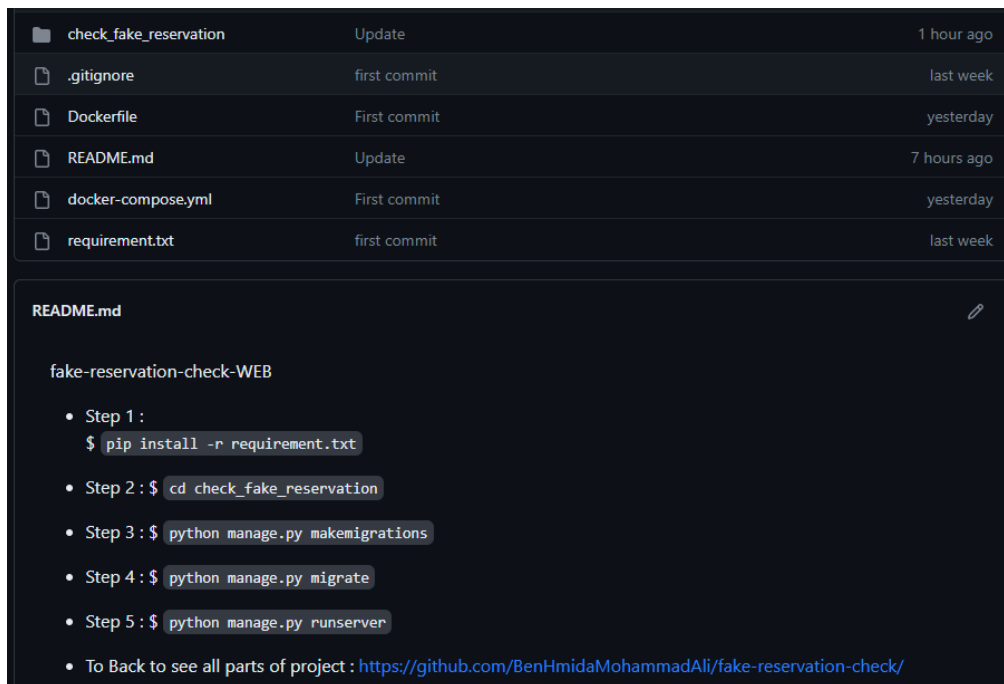


Figure 11 Screenshot of web and deployment Project repository named “Fake-reservation-check-WEB”

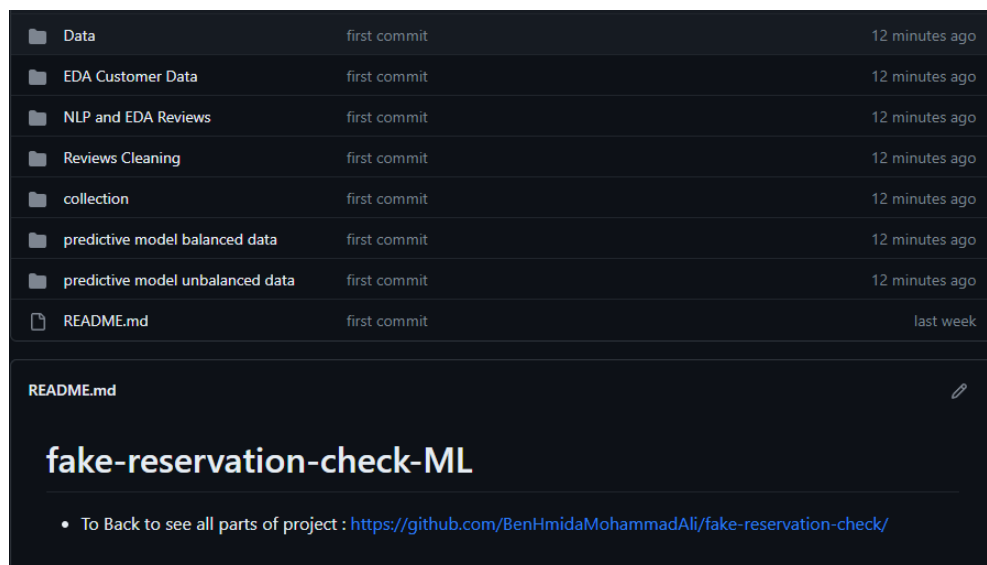


Figure 12 Screenshot of ML Project repository named “Fake-reservation-check-ML”

1.7 CRISP-DM for Machine Learning

CRISP-DM (Cross-Industry Standard Process for Data Mining) is a popular methodology for developing machine learning models. It consists of six phases:

Business understanding: In this phase, the problem to be solved is defined and the project objectives are established.

Data understanding: In this phase, the data is collected and analyzed to identify patterns, trends, and potential issues.

Data preparation: In this phase, the data is cleaned, transformed, and organized to prepare it for analysis.

Modeling: In this phase, the machine learning model is selected and trained on the data.

Evaluation: In this phase, the model is evaluated on a new dataset to assess its performance and identify areas for improvement.

Deployment: In this phase, the model is integrated into a production environment and made available for use.

1.8 Conclusion

In this chapter, we introduced the background and objectives of the project, as well as the problem statement and definitions of machine learning and Business Intelligence. We also discussed the importance of web scraping and data sources in collecting data for the project. In the next chapter, we will discuss the methodology used in this project, including the steps taken to collect, clean, and analyze the data.

Chapter 2: Business Intelligence (BI)

2.1 Definition and Quick Differentiation of BI

Business Intelligence (BI) refers to the tools, techniques, and applications used to collect, integrate, analyze, and present business data. BI solutions are designed to help organizations make data-driven decisions and gain actionable insights into their operations. The primary objective of BI is to provide business users with timely and accurate information that can help them make informed decisions.

The BI process consists of several steps, including data collection, data integration, data analysis, and data presentation. Each step in the BI process requires specialized skills and tools to ensure that the data is accurate, consistent, and useful to decision-makers.

2.2 Data Source

The first step in the BI process is data collection, which involves gathering data from various sources and storing it in a centralized location. In this project, we used two different data sources: Kaggle and British flight reviews.

The Kaggle dataset contained information on airline passenger satisfaction ratings, including attributes such as in-flight service, check-in, onboard experience, and arrival and departure procedures. This dataset was collected from various airlines worldwide, including major and low-cost carriers.

The second data source was British flight reviews, which we obtained by web scraping online review websites. This dataset contained detailed information on flight routes, flight numbers, flight durations, flight frequencies, and flight ratings.

Both datasets were collected in Excel format, and we used Excel to store and manipulate the data before importing it into our data warehouse.

Website from where we scrapped our data:

https://www.airlinequality.com/airline-reviews/british-airways/page/1/?sortby=post_date%3ADesc&pagesize=100

```

1 for i in range(1, 36):
2     page = requests.get(f"https://www.airlinequality.com/airline-reviews/british-airways/page/{i}?sortby=post_date%3ADesc&pagesize=100")
3
4     soup = BeautifulSoup(page.content, "html5")
5
6     for item in soup.find_all("div", class_="text_content"):
7         reviews.append(item.text)
8
9     for item in soup.find_all("div", class_="rating-10"):
10        try:
11            rates.append(item.span.text)
12        except:
13            print(f"Error on page {i}")
14            rates.append("None")
15
16    #date
17    for item in soup.find_all("time"):
18        date.append(item.text)
19
20    #country
21    for item in soup.find_all("h3"):
22        country.append(item.span.next_sibling.text.strip(" "))

```

Error on page 29

Figure 13 Screenshot of part of Web Scraping Code

2.3 Data Preparation

Data preparation involves transforming raw data into a format suitable for analysis. This process typically includes data cleaning, data integration, and data transformation.

In our project, we performed several data preparation tasks, including:

Data cleaning: We removed any duplicates, missing values, or irrelevant data from our datasets.

Data integration: We merged the two datasets using a common key, which was the flight route.

Data transformation: We converted the data into a format suitable for our data warehouse and BI tools. (Set values with type string and object to values of type integer)

2.4 Creating Database Objects

After the data preparation phase, we created the necessary database objects for our data warehouse. This involved designing the schema, creating tables, and defining relationships between them using **SQL Server data Tools**.

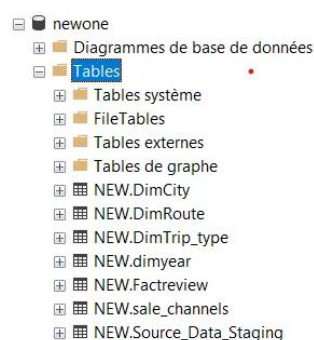


Figure 14 Screenshot of newone database

2.4.1 ETL Source to Staging

We then used ETL (extract, transform, load) processes to transfer the data from the source systems to the staging area of our data warehouse. This involved extracting the data from the Excel files, transforming it to fit the data warehouse schema, and loading it into the staging area.



Figure 15 Graph explained ETL Source to Staging

In this step we created in database the staging table as the next figure.

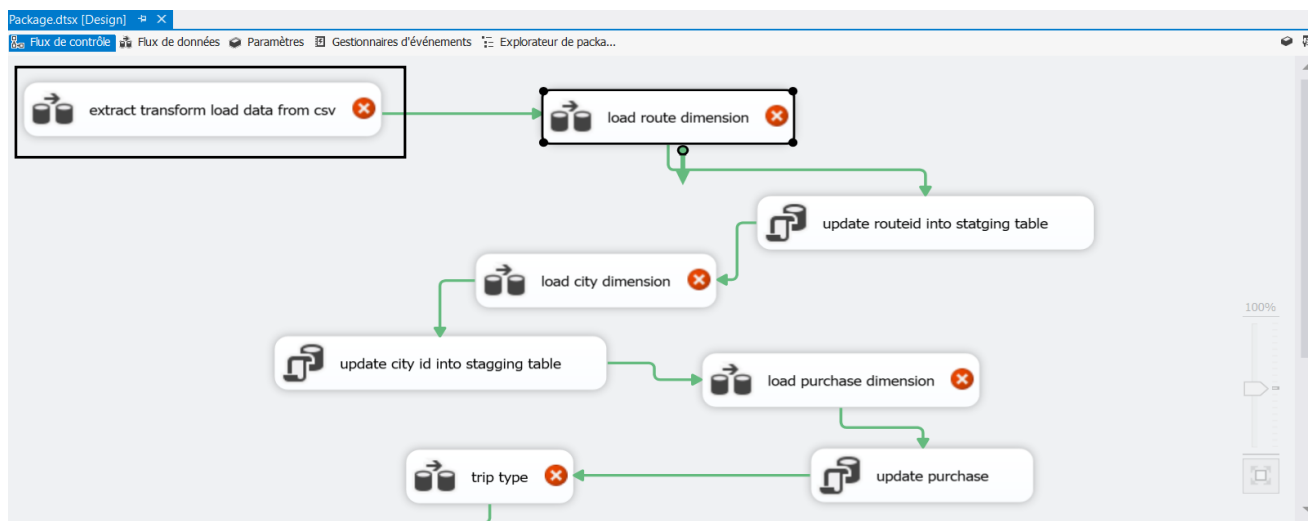


Figure 16 Screenshot of the flow of charging from csv file to staging table

2.4.2 ETL Staging to DW

Next, we used ETL processes to move the data from the staging area to the data warehouse proper. This involved extracting the data from the staging area, transforming it further if necessary, and loading it into the appropriate tables in the data warehouse.



Figure 17 Graph explained ETL staging to Datawarehouse

In this step we created in data warehouse the dim and fact tables as the next figure.

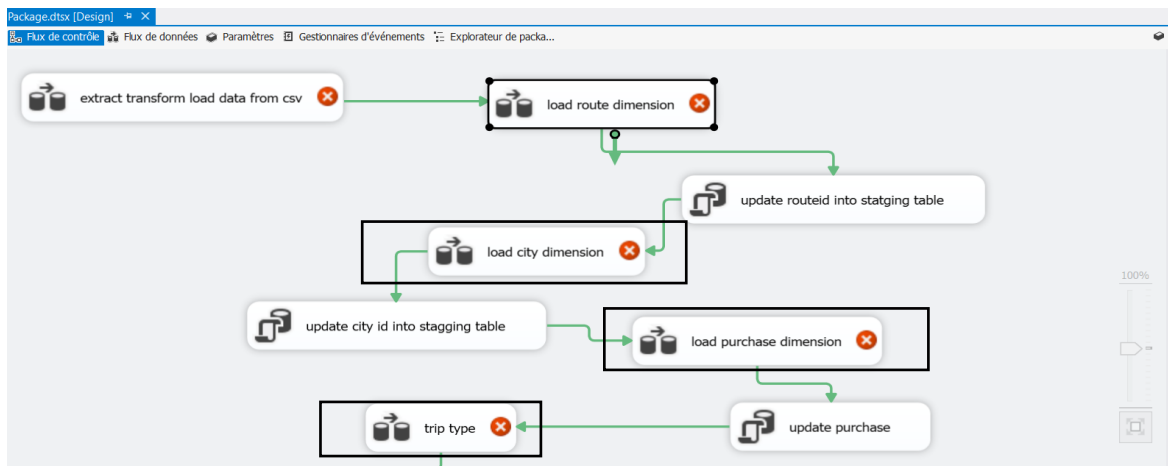


Figure 18 Screenshot of the flow of charging from staging table to Data warehouse 1

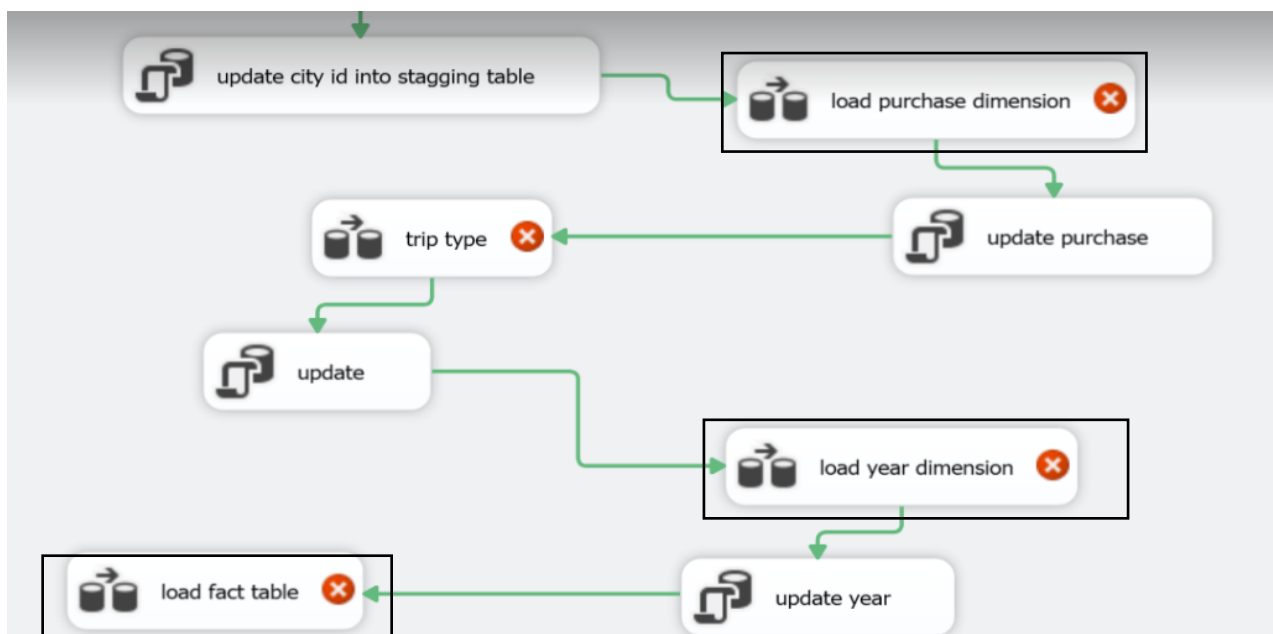


Figure 19 Screenshot of the flow of charging from staging table to Data warehouse 2

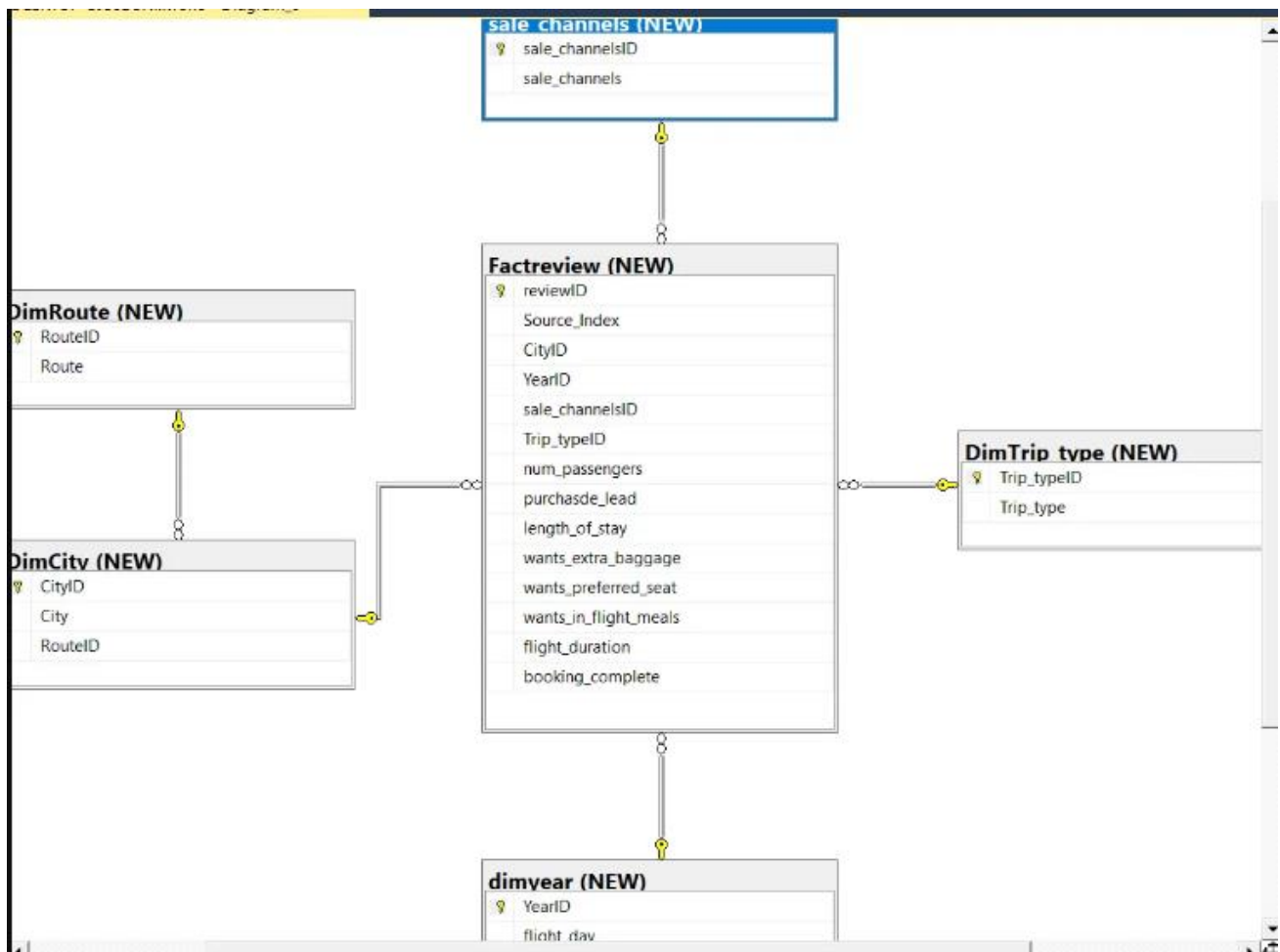


Figure 20 Screenshot of the data warehouse

2.5 Semantic Layer

The semantic layer is a layer of abstraction between the physical data model and the business user. It's implemented using **SQL Server Integration Services (SSIS)** for Tabular and Multidimensional and **SQL Server Management Studio (SSMS)** to creating the Cube.

2.5.1 SSIS Tabular and Multidimensional

SSIS is a Business Intelligence tool from Microsoft that provides OLAP (Online Analytical Processing) and Data Mining functionality for business users. It has two main modes: Tabular and Multidimensional. In the Tabular mode, the data model is created using a relational model and is optimized for in-memory processing. In the Multidimensional mode, the data model is created using a dimensional model and is optimized for disk-based processing.

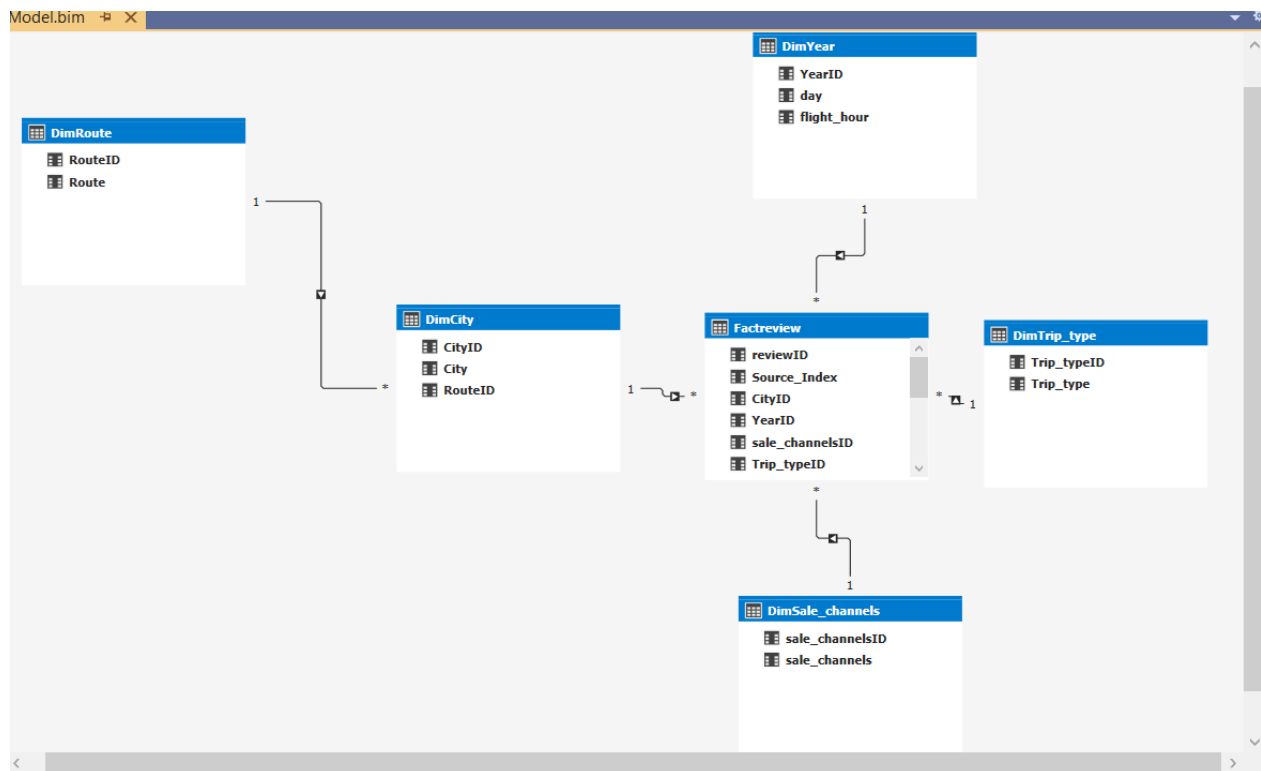


Figure 21 Screenshot of Tabular project using SSIS

2.5.1 SSMS Cube

SSMS Cube, or SQL Server Management Studio Cube, refers to a multidimensional data structure created and managed within SQL Server Management Studio. It allows for efficient storage, analysis, and reporting of multidimensional data by organizing and aggregating information along multiple dimensions.

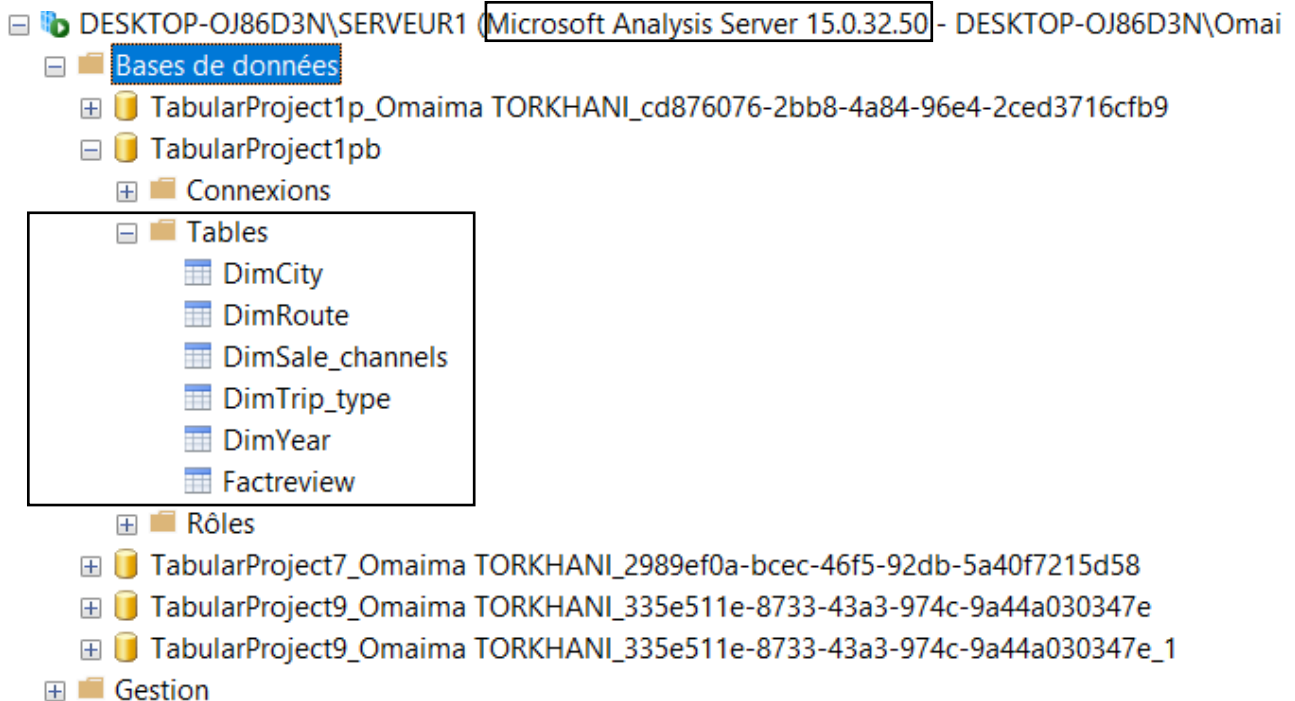


Figure 22 Screenshot of Cube using SSMS

2.6 Creating Power BI Report

Power BI is a business analytics tool from Microsoft that provides interactive visualizations and business intelligence capabilities with an interface simple enough for end users to create their own reports and dashboards. Creating a Power BI report involves connecting to the data source, selecting the appropriate data elements, and designing the report layout. Power BI provides a range of visualizations, including charts, tables, and maps, that can be customized to meet the needs of the user.

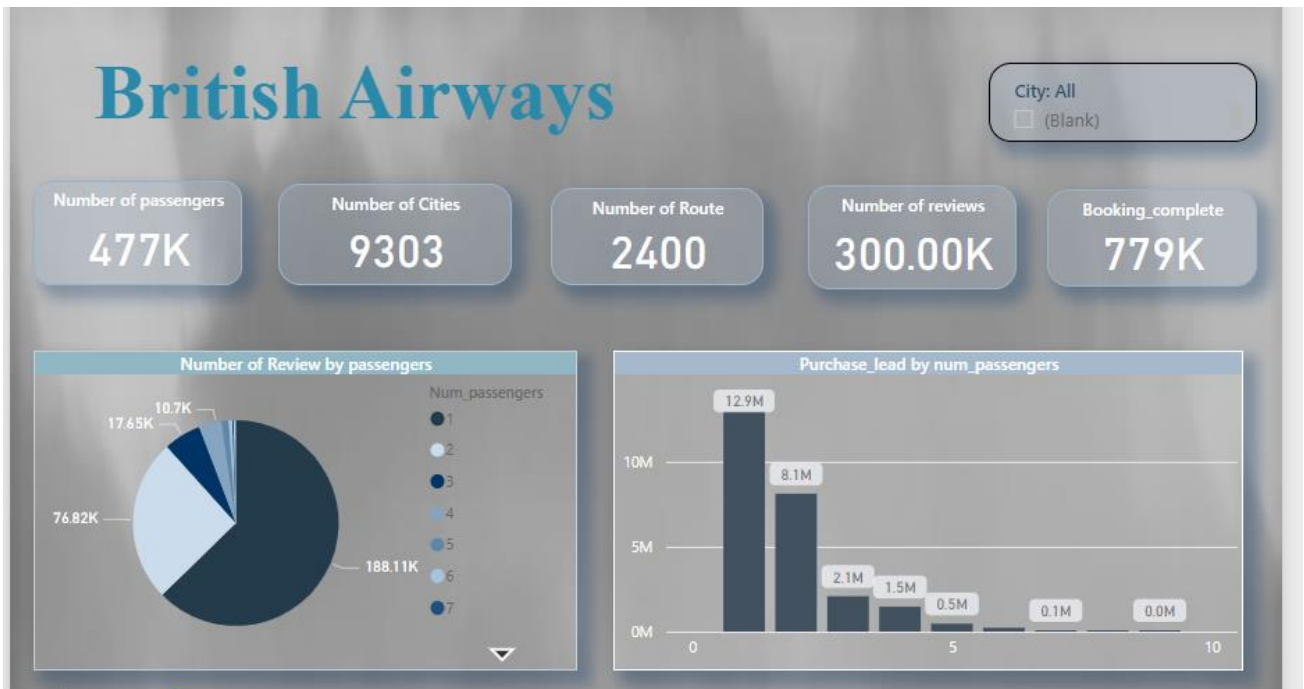


Figure 23 Screenshot of Power BI report 1

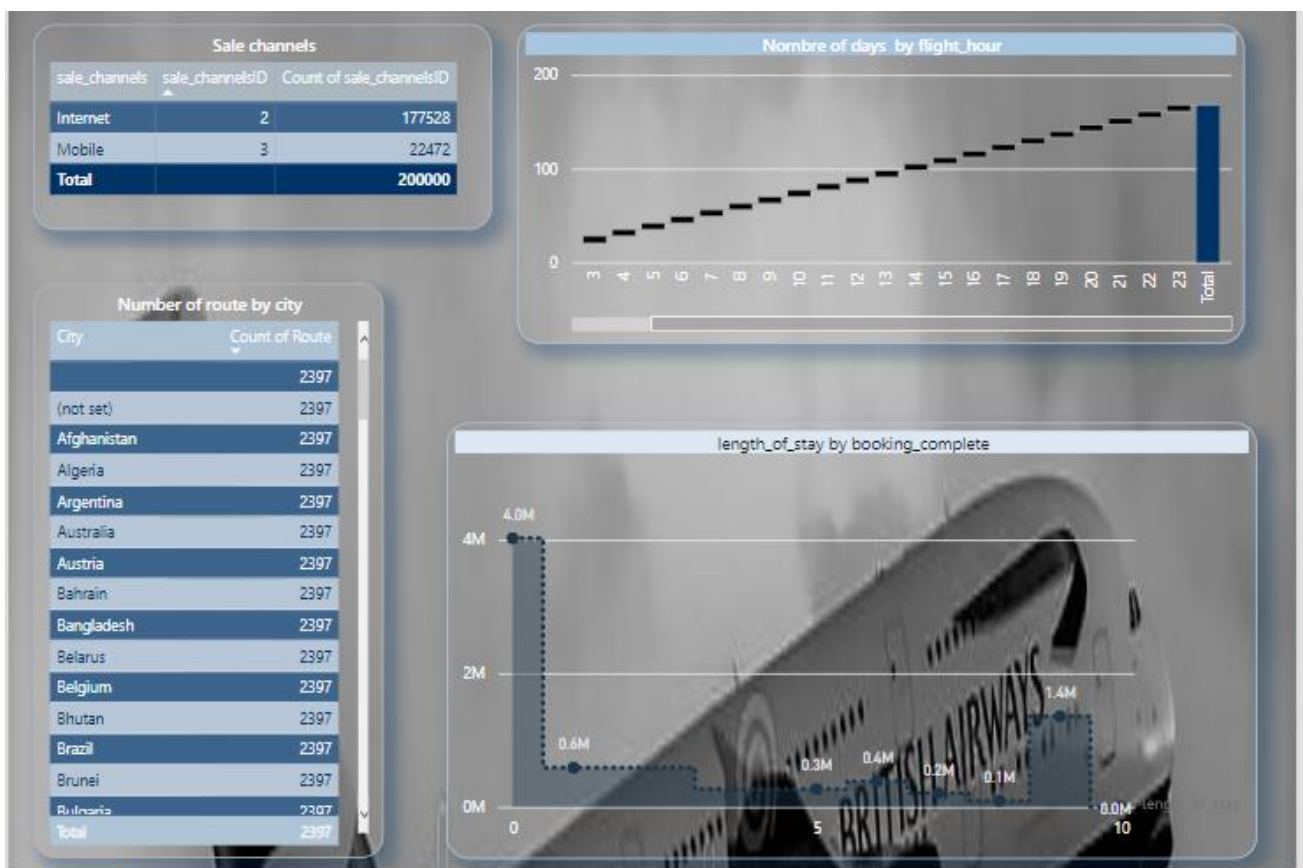


Figure 24 Screenshot of Power BI report 2



Figure 25 Screenshot of Power BI report 3

2.7 Conclusion

In conclusion, Business Intelligence is a critical process that enables organizations to transform data into actionable insights. It involves a range of activities, including data source identification, data preparation, semantic layer development, report authoring, and presentation. Effective Business Intelligence requires a collaborative effort between the Business Intelligence team and the business stakeholders to ensure that the solution meets the needs of the business. In the next chapter, we will discuss the Data Mining process, which is a key component of Business Intelligence.

Chapter 3: Machine learning (ML)

3.1 Definition and Quick Differentiation of ML

Machine learning is a subfield of artificial intelligence that focuses on the development of algorithms and statistical models that enable computer systems to improve their performance on a specific task by learning from data, without being explicitly programmed. The process of machine learning involves feeding a large amount of data to a model, which learns from the data and improves its performance over time.

3.2 Steps for Developing a Machine Learning Model

As we told in first chapter we will use CRISP-DM as method to develop the machine learning model with those steps:

3.2.1 Business understanding

In this phase, the problem to be solved is defined the fake reservations that passengers created for reasons of scams. In this project we had as objectives to create a model using Machine learning algorithm which is based on the dataset to predict the frauds reservations.

3.2.2 Data understanding

In this phase, the data is collected and analyzed to identify patterns, trends, and potential issues. As we are already saw in the previous chapter using BI presentations to know the meaning of features.

In this table (**Table1**) we will know more about meaning of each column in our dataset.

Table 1 Table of meaning of columns

Column name	Description
num_passengers	number of passengers travelling
sales_channel	sales channel booking was made on
trip_type	trip Type (Round Trip, One Way, Circle Trip)
purchase_lead	number of days between travel date and booking date
length_of_stay	number of days spent at destination
flight_hour	hour of flight departure
flight_day	day of week of flight departure
route	origin -> destination flight route
booking_origin	country from where booking was made
wants_extra_baggage	if the customer wanted extra baggage in the booking

wants_preferred_seat	if the customer wanted a preferred seat in the booking
wants_in_flight_meals	if the customer wanted in-flight meals in the booking
flight_duration	total duration of flight (in hours)
booking_complete	flag indicating if the customer completed the booking

Table 2 Divide the columns into features and labels

Features	Label (Target)
num_passengers, sales_channel, trip_type, purchase_lead, length_of_stay, flight_hour, flight_day, route, booking_origin, wants_extra_baggage, wants_preferred_seat, wants_in_flight_meals, flight_duration	booking_complete

3.2.3 Data preparation

In this phase, the data is cleaned, transformed, and organized to prepare it for analysis.

We will follow these 8 steps:

- 1- See more details about our feature

```
In [6]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 14 columns):
#   Column                Non-Null Count  Dtype
---  -
0   num_passengers         50000 non-null   int64
1   sales_channel          50000 non-null   object
2   trip_type              50000 non-null   object
3   purchase_lead          50000 non-null   int64
4   length_of_stay         50000 non-null   int64
5   flight_hour            50000 non-null   int64
6   flight_day             50000 non-null   object
7   route                  50000 non-null   object
8   booking_origin         50000 non-null   object
9   wants_extra_baggage    50000 non-null   int64
10  wants_preferred_seat   50000 non-null   int64
11  wants_in_flight_meals  50000 non-null   int64
12  flight_duration        50000 non-null   float64
13  booking_complete       50000 non-null   int64
dtypes: float64(1), int64(8), object(5)
memory usage: 5.3+ MB
```

Figure 26 Screenshot of dataset details

2- Encoder categorical columns

- For **sales_channel**: if value is Internet so 0 and if value is Mobile so 1
- For **trip_type** if value is RoundTRip so 0, if value is OneWayTrip so 1 and if CircleTrip so 2

```
from sklearn.preprocessing import OneHotEncoder

#create instance of one hot encoder
encoder = OneHotEncoder(handle_unknown='ignore')

#one hot encode Sales Channel
encoder_df = pd.DataFrame(encoder.fit_transform(df[["sales_channel"]]).toarray())
encoder_df = encoder_df.rename(columns={0:'Internet', 1:'Mobile'})
df_final = df_final.join(encoder_df)

#one hot encode trip type
encoder_df = pd.DataFrame(encoder.fit_transform(df[["trip_type"]]).toarray())
encoder_df = encoder_df.rename(columns={0:'RoundTRip', 1:'OneWayTrip', 2:'CircleTrip'})
df_final = df_final.join(encoder_df)
```

Figure 27 Code of Encoder function

3- Delete categorical columns

```
#drop categorical columns now
df_final.drop(['sales_channel', 'trip_type', 'booking_origin', 'route'], axis=1, inplace = True)
```

Figure 28 Code of drop categorical columns

4- Create Label dataframe

```
#store the Label for supervised Learning
label = df['booking_complete']
```

Figure 29 Code of Creating Label dataframe

5- Delete Label column from features dataframe

```
: df_final = df_final.drop('booking_complete', axis=1)
```

Figure 30 Code of deleting Label from features dataframe

6- Normalizing the values

Normalizing the values in a dataset is a common preprocessing step in machine learning. It involves scaling the values of the features to a standard range, usually between 0 and 1 or -1 and 1, to ensure that they are on a similar scale

```

from sklearn.preprocessing import StandardScaler

#create a standard scaler object
scaler = StandardScaler()

#fit and transform the data
scaled_df = scaler.fit_transform(df_final)

#create a dataframe of scaled data
scaled_df = pd.DataFrame(scaled_df, columns = df_final.columns)

# add the labels back to the dataframe
scaled_df['label'] = label

```

Figure 31 Code of Normalizing the values

7- Correlation matrix

A correlation matrix is a table that shows the correlation coefficients between variables in a dataset. The coefficients range from -1 to 1 and indicate the strength and direction of the relationship between variables.

1 indicates a strong positive correlation.
 0 indicates no correlation.
 -1 indicates a strong negative correlation.

By examining the matrix, you can quickly identify patterns and relationships between variables.

```

corr = scaled_df.corr()

plt.figure(figsize=(10,7))

#plot the heatmap
sns.heatmap(corr)

```

Figure 32 Code to create the correlation matrix

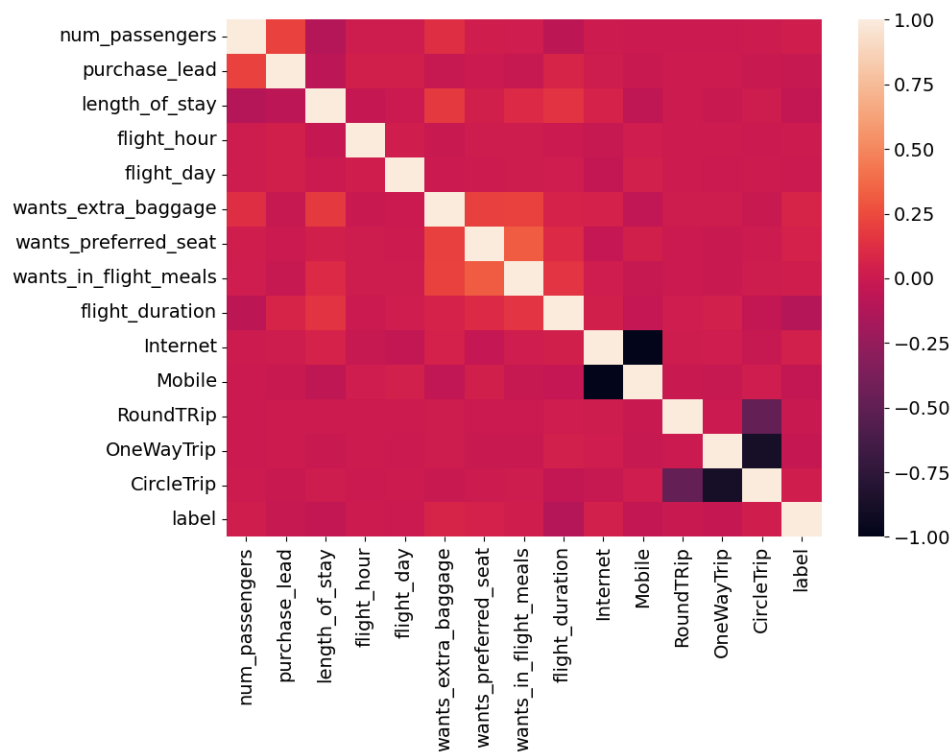


Figure 33 figure of correlation matrix

8- Balancing our data

```
scaled_df.label.value_counts()

0    42506
1     7476
Name: label, dtype: int64

#create a dataframe having all Labels 0 with 10000 samples
scaled_df_0 = scaled_df[scaled_df.label == 0].sample(n=8000)

#concatenate the two dataframes, one having all Labels 0 and other having all Labels as 1
scaled_df_new = pd.concat([scaled_df[scaled_df.label == 1], scaled_df_0], ignore_index=True)

#shuffle the dataframe rows
scaled_df_new = scaled_df_new.sample(frac = 1).reset_index(drop=True)
```

Figure 34 Code of Balancing the dataset

9- Find the important features

In this step we will find the most important features that the model used to predict the target. (List of 5 most important features for our model).

purchase_lead,
length of stay,
flight_hour,
flight duration,
flight_day

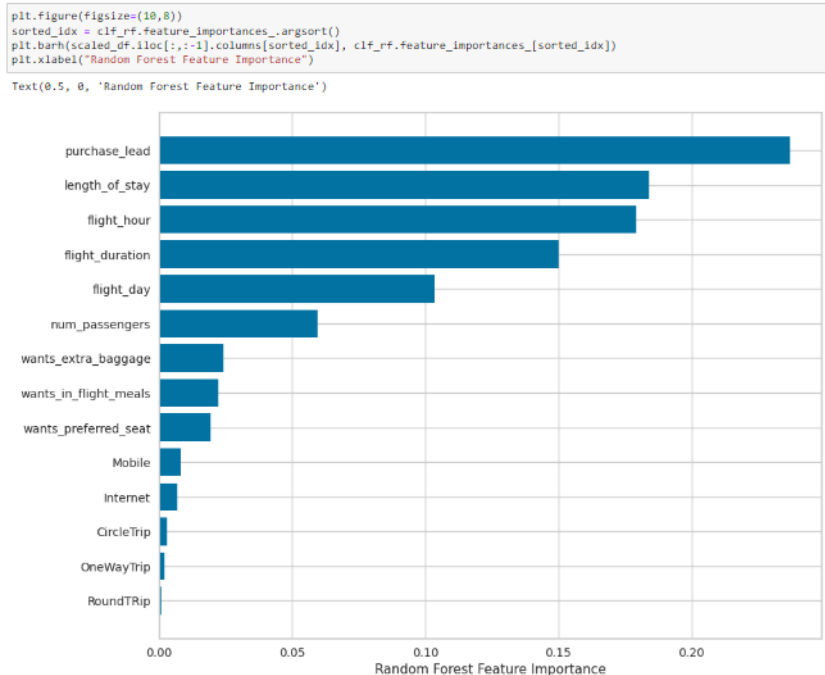


Figure 35 Code and graph representation of the most important features

3.2.4 Modeling

In this phase, the machine learning model is selected and trained on the data.

1- Splitting Train and Test Data

```
: from sklearn.model_selection import train_test_split
X = scaled_df.iloc[:, :-1]
y = scaled_df['label']
X_train, X_test, y_train, y_test = train_test_split(X.to_numpy(), y.to_numpy(), test_size=0.20, random_state=42)
```

Figure 36 Code of splitting data into test and train data

2- Model creation

(In this part we are already try other models to use but when we evaluated, we find that this model had best scores).

In this step we will create an instance of the classifier and fit the training data. We choose **Random Forest Classifier** algorithm

We choose this algorithm for:

Accuracy: It delivers high accuracy in predicting outcomes for both classification and regression tasks.

Feature importance: It provides a measure of feature importance, aiding in feature selection and understanding data relationships.

Overfitting reduction: The ensemble nature of Random Forests helps reduce overfitting by averaging predictions.

Scalability: It efficiently handles large datasets and performs well in parallel computing environments.

```
: clf_rf = RandomForestClassifier(max_depth =50 , min_samples_split=5,random_state=0)

: y_pred_train = model_fit_predict(clf_rf, X_train, y_train, X_train)
  set(y_pred_train)
```

Figure 37 Code of Creation random forest model and the training

- 3- As the you can find in the next step (Evaluation) we find that the data base is unbalanced data with (42506 with 0 labels and 7476 with 1 labels)

```
scaled_df.label.value_counts()

0      42506
1       7476
Name: label, dtype: int64
```

Figure 38 Screenshot of labels values

- 4- After balancing our data, we train our model to new dataset.

```
X = scaled_df_new.iloc[:, :-1]
y = scaled_df_new['label']

X_train, X_test, y_train, y_test = train_test_split(X.to_numpy(), y.to_numpy(), test_size=0.20, random_state=42)

#create an instance of the classifier and fit the training data
clf_rf = RandomForestClassifier(n_estimators=50,max_depth =50 , min_samples_split=5,random_state=0)

cm = ConfusionMatrix(clf_rf, classes=[0,1])
cm.fit(X_train, y_train)
```

Figure 39 Code of new model with new balanced dataset

3.2.5 Evaluation

In this phase, the model is evaluated on a new dataset to assess its performance and identify areas for improvement.

- 1- Create functions to calculate the accuracy, precision and f1 scores to evaluate the model.


```

def acc_score(y_true, y_pred):
    return accuracy_score(y_true, y_pred)

def pre_score(y_true, y_pred):
    return precision_score(y_true, y_pred)

def f_score(y_true, y_pred):
    return f1_score(y_true, y_pred)

```

Figure 40 Code of evaluation score functions

2- Test our model

- Our first model gives us a training test:

Accuracy = 0.93
Precision = 1.0
f1-score = 0.72

```

#f1 score for training data
f1 = round(f1_score(y_train, y_pred_train),2)

#accuracy score for training data
acc = round(accuracy_score(y_train, y_pred_train),2)

#precision score for training data
pre = round(precision_score(y_train, y_pred_train),2)

print(f"Accuracy, precision and f1-score for training data are {acc}, {pre} and {f1} respectively")

```

Accuracy, precision and f1-score for training data are 0.93, 1.0 and 0.72 respectively

Figure 41 Code of testing and evaluating the model

- In this step we tested our data and find that data is not balanced (see figure36)
⇒ That make our model in case of overfitting.
- After balancing our data, we find that our model had scores of

Accuracy = 0.63
Precision = 0.62
Recall = 0.59
f1-score = 0.6

```

y_pred_test = model_fit_predict(clf_rf, X_train, y_train, X_test)

#f1 score for training data
f1 = round(f1_score(y_test, y_pred_test),2)

#accuracy score for training data
acc = round(accuracy_score(y_test, y_pred_test),2)

#precision score for training data
pre = round(precision_score(y_test, y_pred_test),2)

recall = round(recall_score(y_test, y_pred_test),2)

specificity = round(recall_score(y_test, y_pred_test, pos_label=0),2)

print(f"Accuracy, precision, recall and f1-score for training data are {acc}, {pre}, {recall}, {specificity} and {f1} respectively")

```

Figure 42 Code of new model test

3.2.6 Deployment

In this phase, the model is integrated into a production environment and made available for use. We take the next chapter to show the application that we developed.

3.3 Conclusion

To summarize, this chapter outlined the development and evaluation of our Machine Learning (ML) model using the CRISP-DM methodology. By following this structured approach, we successfully addressed the business problem through stages like data collection, preprocessing, model training, and evaluation. The CRISP-DM framework ensured a systematic and well-documented process, resulting in a reliable ML model. The insights gained from this chapter form the basis for the subsequent sections of the report, including discussing the results, limitations, and future improvements of the ML model.

Chapter 4: Deployment

4.1 Definition and Quick Differentiation of ML Deployment

Machine Learning (ML) deployment refers to the process of integrating and running trained ML models in real-world applications or systems to make predictions or decisions based on new input data. It involves taking the model that was developed and trained offline and making it available for practical use. Django, on the other hand, is a popular high-level Python web framework that simplifies the development of web applications. It provides a set of tools and functionalities for building robust and scalable web applications quickly.

4.2 Django

Django is a popular Python web framework that simplifies web application development. It provides a set of tools and functionalities that help developers build web applications quickly and efficiently. At its core, Django follows the Model-View-Controller (MVC) architectural pattern, where the "Model" represents the data structure and database schema, the "View" handles the logic for rendering the user interface, and the "Controller" manages the flow of data between the Model and View.

4.3 Web Application with Django

In our web application admin can add flights and see reservation of passenger and see their details.

Figure39 show the project architecture.

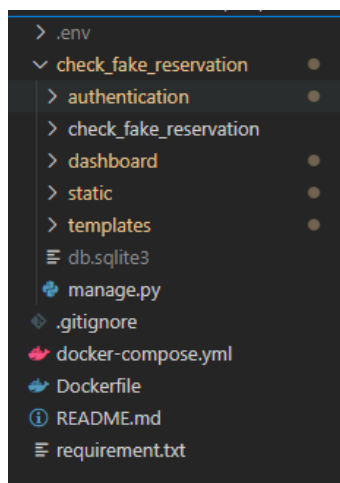


Figure 43 Screenshot of Project Architect

Figure40 show the login page, if user authenticated as superuser he will get an admin dashboard as **figure41** which in can manage passengers, flights and passenger's reservations. If user authenticated as simple user he will get a passenger dashboard as **figure42** which can see available flights and reserve one for his trip.

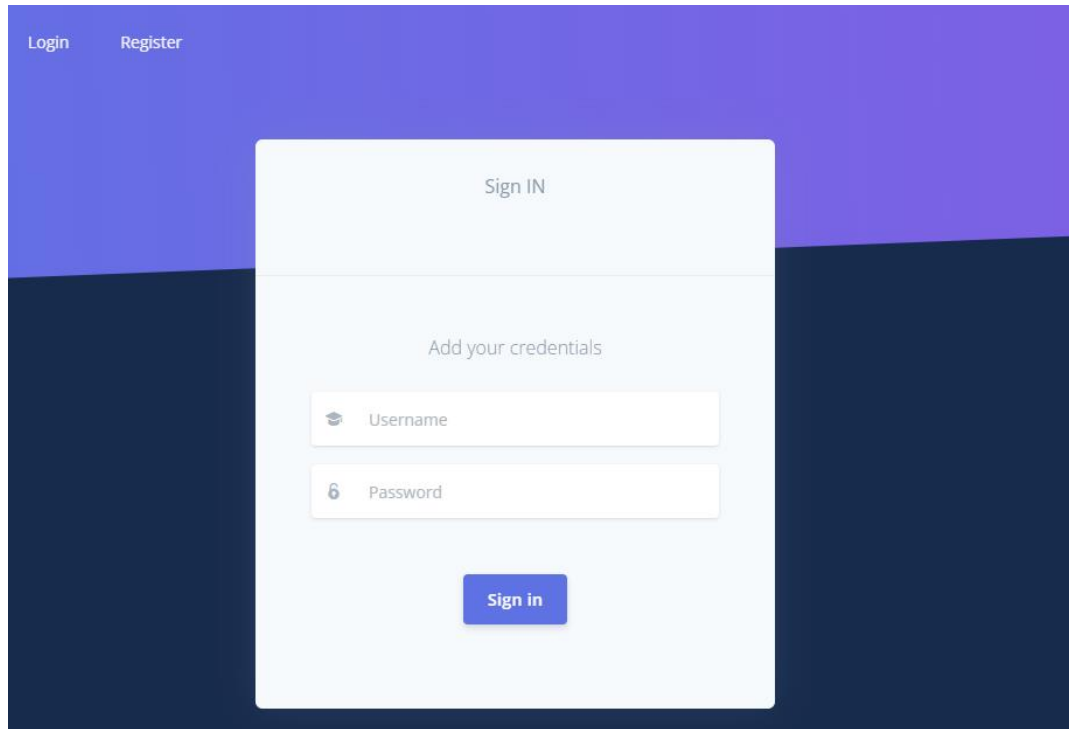


Figure 44 Screenshot of Login Page

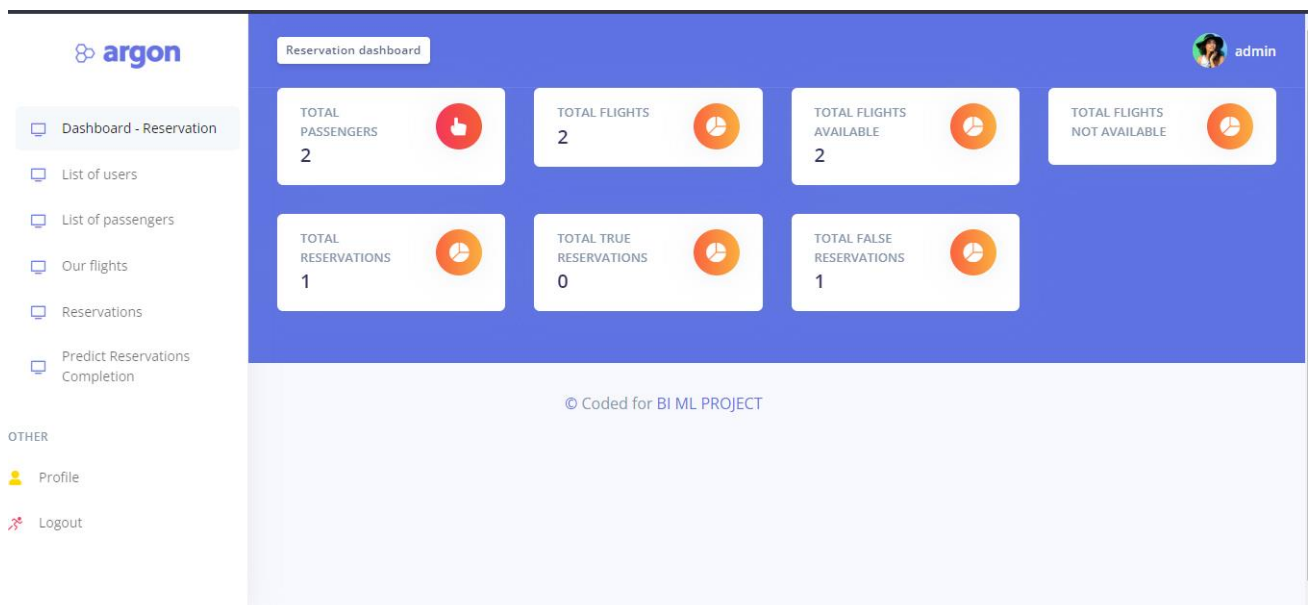


Figure 45 Screenshot of Dashboard Admin Page

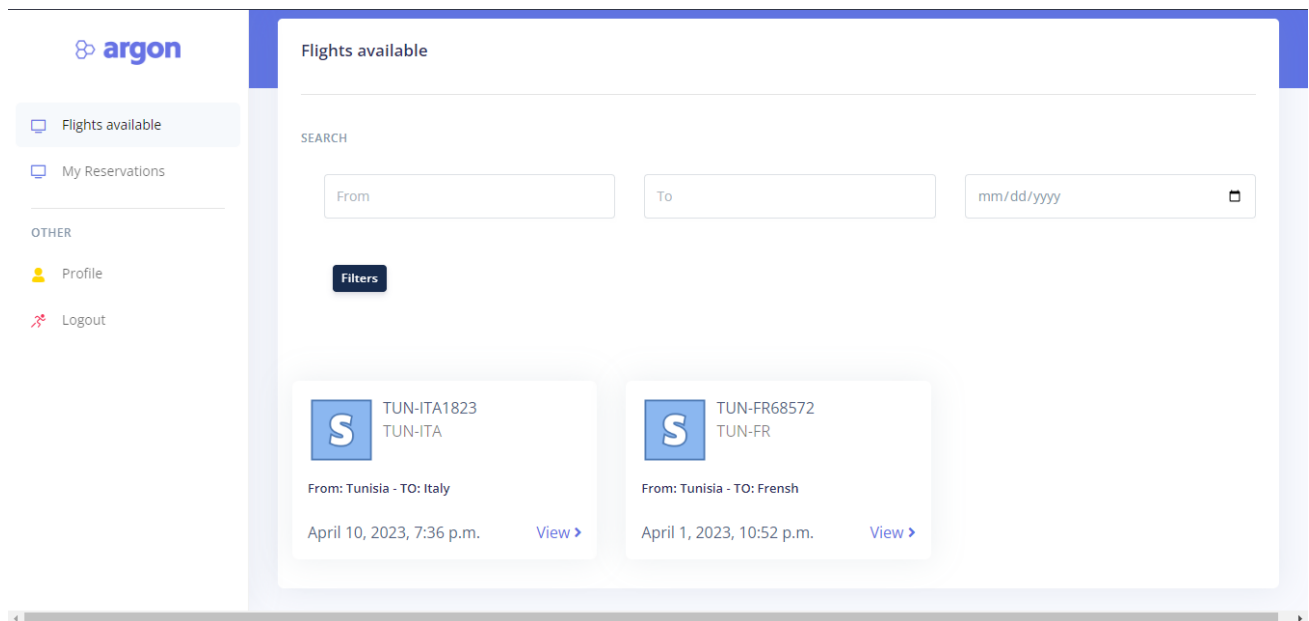


Figure 46 Screenshot of Dashboard simple user Page

4.5 ML Model - Deployment

To add the ML model in our application we will add a new folder in our Django project for the trained model.

1- Using pickle to create ml_model.plk and scaler.plk

Pickle: To save and load a saved model from a Pickle file, all you need to do is pass the “pickled” model into the Pickle load () function and it will be deserialized.

After we training models and choosing the best one with most score evaluation, we create “ml_model.plk” to save our model and use it on our web application and “scaler.plk” to save model to normalize the values.

```
import pickle
pickle.dump(cm , open("ml_model.pkl","wb"))
```

Figure 47 Code using pickle module to creation our ml_model.plk

```
: import pickle
pickle.dump(scaler , open("scaler.pkl","wb"))
```

Figure 48 Code using pickle module to creation our scaler.plk

ml_model.plk and scaler.plk file will be created in our project

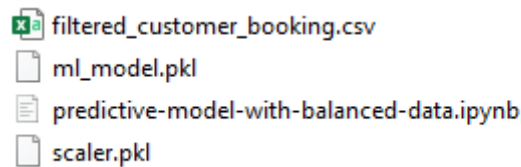


Figure 49 Screenshop of the new files created

2- Create new folder for model method

- Create folder model that had sav file and model.py that's in we will create the predict function using the model

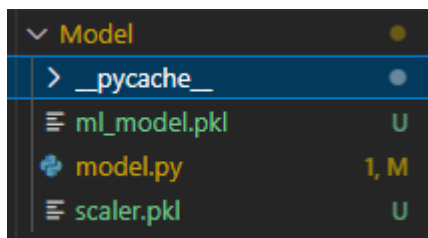


Figure 50 Screenshot of the Model folder architect

- In model.py we will create a function that get in parameter the features and return the predicted values

```
import pickle
import numpy as np

def model_predict(L):
    model = pickle.load(open('C:/Users/Administrator/Desktop/fake-reservation-check/fake-reservation-check-WEB/che
    scaled = pickle.load(open('C:/Users/Administrator/Desktop/fake-reservation-check/fake-reservation-check-WEB/ch

    d = np.array(L)

    # Reshape the input data to have 14 features
    x = d.reshape(1, -1)
    prediction = model.predict(scaled.transform([
        d
    ]))

    if prediction == 0:
        return 'Fake'
    elif prediction ==1 :
        return 'True'
    else :
        return 'Error'
```

Figure 51 Code of model predict function

- In view function of predict button we will update list of reservation depend of the return value of model predict function.

```
def predict_reservation (request):
    reservation_predicted_list = ReservationFlightPredictions.objects.all()

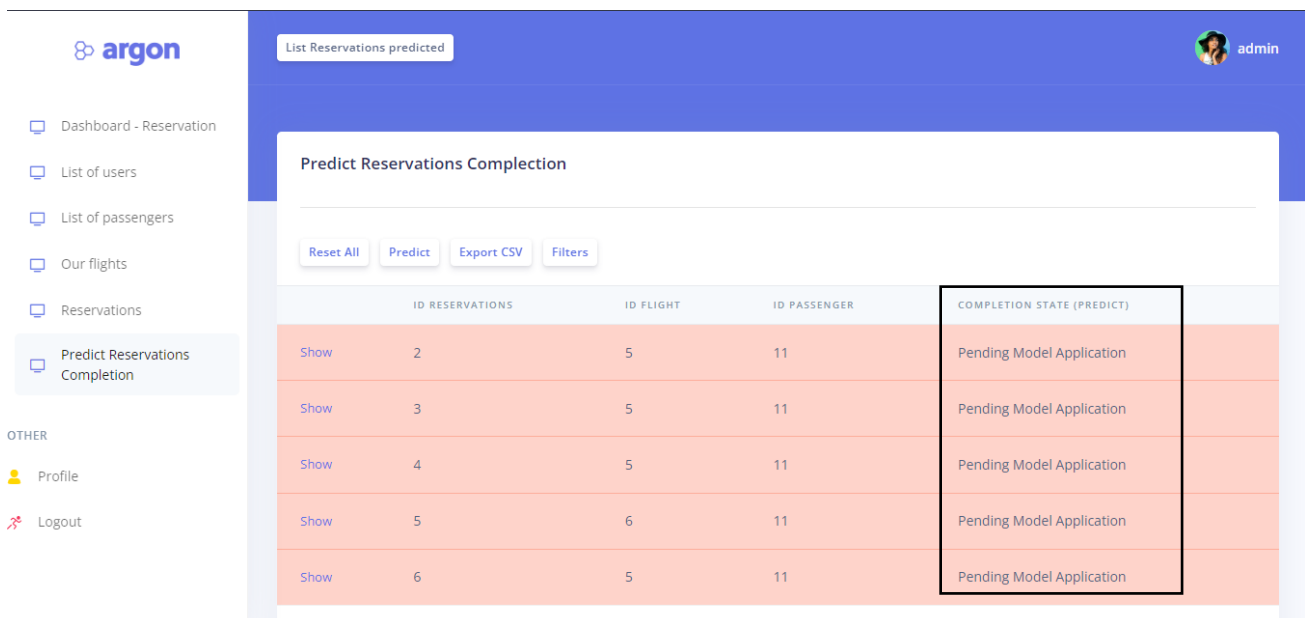
    for i in reservation_predicted_list:
        L= [i.reservationFlight.number_of_chairs_to_reserve, i.reservationFlight.travel_duration,
            i.reservationFlight.travel_duration,hour,day,i.reservationFlight.extra_baggage,
            i.reservationFlight.preffered_seat,i.reservationFlight.meal,i.reservationFlight.travel_duration,
            Internet,Mobile ,RoundTRip,OneWayTrip,CircleTrip]
        result = model_predict(L)
        i.complete = result
        i.save()

    reservation_predicted_list = ReservationFlightPredictions.objects.all()

    context ={
        'reservation_predicted_list':reservation_predicted_list,
        'segment':'predict_reservations',
        'dashboard_type':'List Reservations predicted',
    }
    return render(request, 'home/predict_reservation/list.html', context)
```

Figure 52 Code of view function predict_reservation code

- 3- For the first step our Django application show us the list of reservations with state “Pending Model Application”

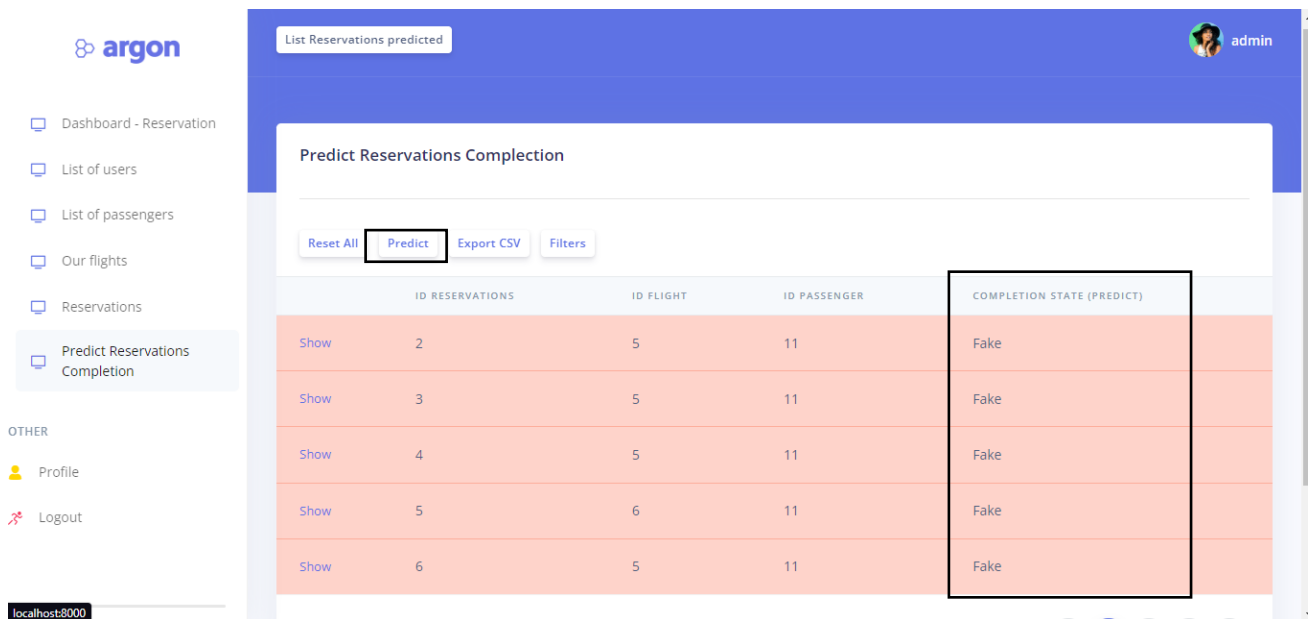


The screenshot shows a Django application interface. On the left is a sidebar with the 'argon' logo and navigation links: Dashboard - Reservation, List of users, List of passengers, Our flights, Reservations, and Predict Reservations Completion (which is highlighted). Below these are 'OTHER' links: Profile and Logout. The main content area has a blue header with 'List Reservations predicted' and a user profile 'admin'. Below the header is a section titled 'Predict Reservations Completion' with buttons for 'Reset All', 'Predict', 'Export CSV', and 'Filters'. A table displays the following data:

	ID RESERVATIONS	ID FLIGHT	ID PASSENGER	COMPLETION STATE (PREDICT)
Show	2	5	11	Pending Model Application
Show	3	5	11	Pending Model Application
Show	4	5	11	Pending Model Application
Show	5	6	11	Pending Model Application
Show	6	5	11	Pending Model Application

Figure 53 Screenshot of a list of reservations awaiting the application of the model.

- 4- For the next step our if we clicked on “predict” button the application will predict list of reservation based on our trained model.

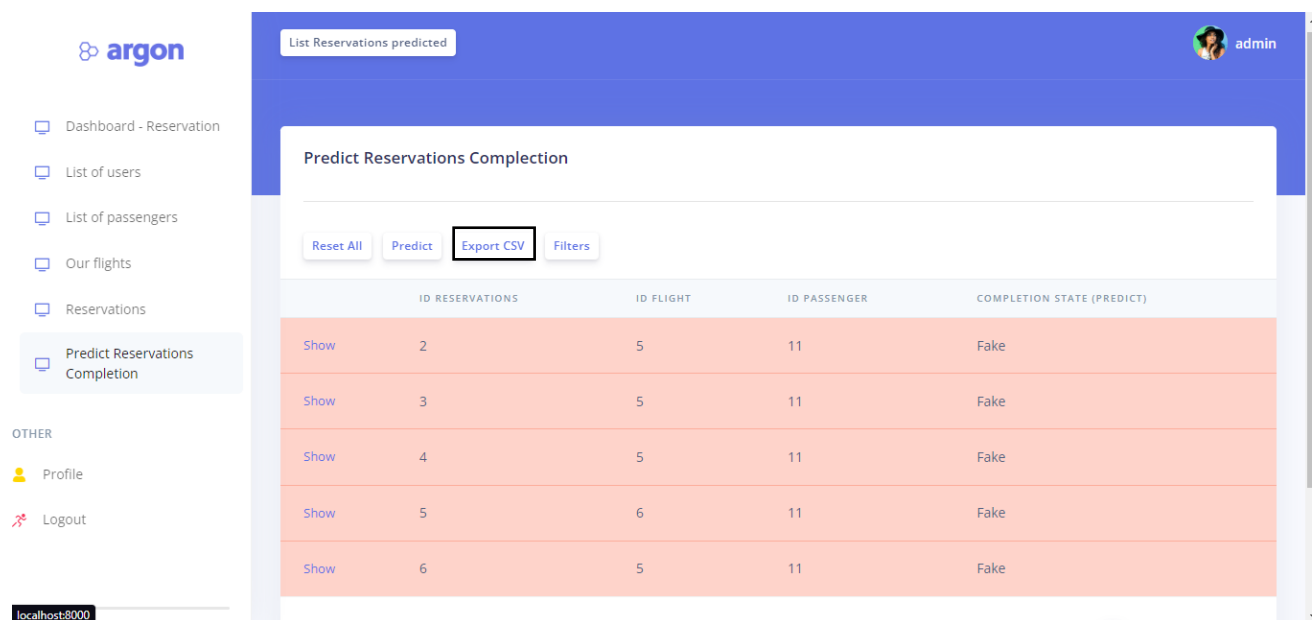


The screenshot shows the Argon application interface. On the left is a sidebar with navigation links: Dashboard - Reservation, List of users, List of passengers, Our flights, Reservations, Predict Reservations Completion, and OTHER (Profile, Logout). The main content area is titled 'List Reservations predicted' and 'Predict Reservations Completion'. It features buttons for 'Reset All', 'Predict' (highlighted), 'Export CSV', and 'Filters'. Below these buttons is a table with the following data:

	ID RESERVATIONS	ID FLIGHT	ID PASSENGER	COMPLETION STATE (PREDICT)
Show	2	5	11	Fake
Show	3	5	11	Fake
Show	4	5	11	Fake
Show	5	6	11	Fake
Show	6	5	11	Fake

Figure 54 Screenshot of a list of reservations after the application of the model


- 5- If the “Export CSV” button is clicked, the application will Create a new csv file that had all the data of predictions



The screenshot shows the Argon application interface, similar to Figure 54, but with the 'Export CSV' button highlighted. The table data remains the same:

	ID RESERVATIONS	ID FLIGHT	ID PASSENGER	COMPLETION STATE (PREDICT)
Show	2	5	11	Fake
Show	3	5	11	Fake
Show	4	5	11	Fake
Show	5	6	11	Fake
Show	6	5	11	Fake

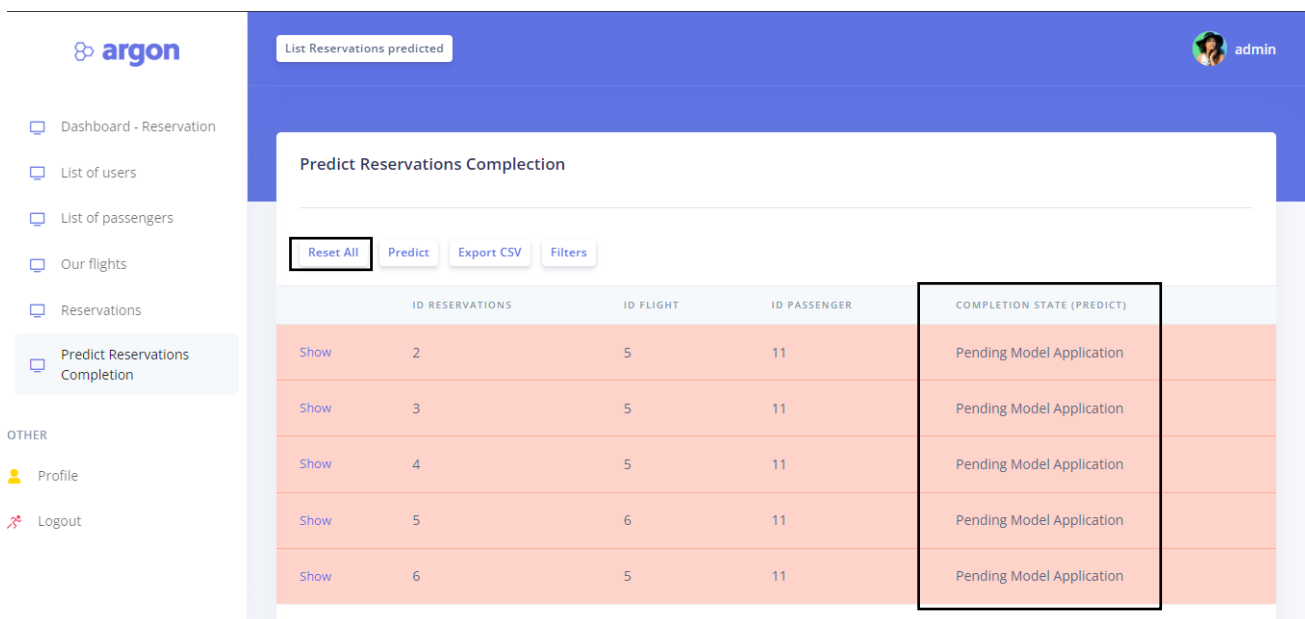
Figure 55 List of prediction with export csv button



	A	B	C	D	E	F
1	ID RESERV	ID FLIGHT	ID PASSEN	COMPLETION STATE (PREDICT)		
2	2	5	11	Fake		
3	3	5	11	Fake		
4	4	5	11	Fake		
5	5	6	11	Fake		
6	6	5	11	Fake		
7						
8						

Figure 56 List of prediction in csv file

- 6- If the “Reset” button is clicked, the application will set the prediction state to " Pending Model Application " to allow the model to be reused for predicting reservations.



argon

List Reservations predicted

admin

Predict Reservations Completion

Reset All Predict Export CSV Filters

	ID RESERVATIONS	ID FLIGHT	ID PASSENGER	COMPLETION STATE (PREDICT)
Show	2	5	11	Pending Model Application
Show	3	5	11	Pending Model Application
Show	4	5	11	Pending Model Application
Show	5	6	11	Pending Model Application
Show	6	5	11	Pending Model Application

Figure 57 Screenshot of a list of reservations after reset the states

4.6 Conclusion

In summary, deploying the machine learning model to an application is important for accessibility, integration, real-time insights and business intelligence. It allows users to seamlessly interact with the tools and take advantage of their functionality within the application environment.

Conclusion

In conclusion, our project aimed to address the issue of fake reservations in the airline industry using Business Intelligence and machine learning techniques.

By analyzing data from British Airways reservation system, we developed a predictive model to identify fraudulent reservations. Through effective data collection, cleaning, and analysis, we were able to gather insights from multiple sources, including Kaggle and web scraping. The utilization of SSAS Tabular and Power BI reports facilitated interactive data exploration and visualization.

The Random Forest Classifier was selected for its accuracy, robustness to noise and outliers, feature importance analysis, and scalability. This model proved effective in predicting the likelihood of fraudulent reservations.

By deploying the developed model in a web application, we enhanced accessibility and user interaction, improving the overall reservation system's security and customer experience. Our project contributes to the airline industry by providing a reliable solution for detecting and eliminating fraudulent activities. Implementing our recommendations can help airlines enhance their reservation systems, build customer trust, and improve fraud detection capabilities. Future work could involve exploring other machine learning algorithms, refining data collection processes, incorporating real-time data feeds, and collaborating with industry stakeholders to implement fraud detection systems on a larger scale.

Overall, our project showcases the power of Business Intelligence and machine learning in addressing critical challenges in the airline industry and continuous improvement in fraud detection and prevention.