

AIRLINE DATA ANALYSIS USING SPARK TECHNOLOGIES

Submitted in partial fulfillment of the requirements
for the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

B.YESHASWI ASHA (17SCS0044)

BHARGAVI MADDALI (17SCS0249)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC JEPPIAAR NAGAR, RAJIV
GANDHI SALAI, CHENNAI – 600 119**

MARCH 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai - 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **B.YEASWISH ASHA(17SCS0044)** and **BHARGAVI MADDALI (17SCS0249)** who carried out the project entitled **"AIRLINE DATA ANALYSIS USING SPARK TECHNOLOGIES"** under my supervision from **December 2020** to **March 2021**.

Internal Guide

Dr. MAHESWARI, M.E., Ph.D.,

Head of the Department

Dr.L.LAKSHMANAN, M.E., Ph.D.,

Dr. S. VIGNESWARI, M.E., PH.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **B.YESHASWI ASHA, BHARGAVI MADDALI** hereby declare that the Project Report entitled “**AIRLINE DATA ANALYSIS USING SPARK TECHNOLOGIES**” is done by me under the guidance of **Dr. Maheswari, M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean, School of Computing, Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Maheshwari, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Our project analyzes the US domestic flights dataset using pyspark dataframes. We aim to predict which flight/flight carrier is most likely to be cancelled or delayed. Using Pyspark, a big data technology tool, we can therefore work with RDDs/Dataframes/Datasets in Python. Our analysis will help us find out what are the top airlines with the maximum delays/cancellations in the US. Also, our project will investigate whether there are any factors that cause flight delays/cancellations. Further, our project predicts if a flight will get cancelled or not. In order to build the predictive system Machine learning models were incorporated, where we modeled a binary classification problem with categorical variables, logistic regression, decision tree classifier, random forest and gradient boosted trees.

TABLE OF CONTENTS

ABSTRACT	v
LIST OF FIGURES	viii

CHAPTER No.	TITLE	PAGE No.
1.	INTERODUCTION	1
	1.1 EXISTING SYSTEM	1
	1.2 PROPOSED SYSTEM	1
	1.3 LITERATURE SURVEY	2
2.	AIM AND SCOPE	3
	2.1 DATA ANALYTICS DEVELOPMENT OF FDR DATA FOR AIRLINE	3
	2.2 BIG DATA ANALYTICS SOCIAL MEDIA	3
	2.3 BIG DATA ANALYTICS IN AIRLINES	4
	2.4 HARDWARE REQUIREMENTS	4
	2.5 SOFTWARE REQUIREMENTS	4
3.	METHODOLOGY	5
	3.1 SOFTWARE ENVIRONMENT	5
	3.1.1 PYTHON	5
	3.1.2 HISTORY OF PYTHON	5
	3.1.3 PYTHON FEATURES	6
	3.1.4 GETTING PYTHON	6
	3.2 FIRST PYTHON PROGRAM	7
	3.2.1 INTERACTIVE MODE PROGRAMMING	7
	3.2.2 SCRIPT MODE PROGRAMMING	8
	3.3 FLASK FRAMEWORK	8
	3.4 MODULES	12
	3.5 DATA PROFILING	13
	3.6 DATA CLEANING AND PREPARATION	13
4.	RESULT AND ANALYSIS	14
	4.1 STATISTICAL DATA ANALYSIS	14
	4.2 MODEL PREDICTION	14
	4.3 DECISION TREE CLASSIFIERS	15

4.4	REQUIREMENT ANALYSIS	15
4.4.1	FUNCTIONAL REQUIREMENT	15
4.4.2	USABILITY	15
4.4.3	ROBUSTNESS	15
4.4.4	SECURITY	15
4.4.5	RELIABILITY	15
4.4.6	COMPATIBILITY	16
4.4.7	FLEXIBILITY	16
4.4.8	SAFETY	16
4.5	SYSTEM DESIGN AND TESTING PLAN	16
4.5.1	INPUT DESIGN	16
4.5.2	OUTPUT DESIGN	16
5.	DESIGN AND TESTING	18
5.1	SYSTEM STUDY	18
5.1.1	FEASIBILITY STUDY	18
5.1.2	ECONOMICAL FEASIBILITY	18
5.1.3	TECHNICAL FEASIBILITY	18
5.1.4	SOCIAL FEASIBILITY	19
5.2	DATA FLOW DIAGRAMS	19
5.3	UML DIAGRAMS	23
5.3.1	USE CASE DIAGRAM	23
5.3.2	SEQUENCE DIAGRAM	24
5.4	SYSTEM TESTING	25
5.4.1	TYPES OF SYSTEM TESTS	26
5.5	TEST STRATEGY AND APPROACH	28
5.5.1	TEST OBJECTIVES	28
5.5.2	FEATURES TO BE TESTED	28
5.6	INTEGRATION TESTING	28
5.7	ACCEPTANCE TESTING	28
6.	CONCLUSION	29
	REFERENCES	30
	A .PLAGARISM REPORT	
	B .PAPER	

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
2.1	ARCHITECTURE DIAGRAM	4
5.1	CONTEXT LEVEL DIAGRAM	20
5.2	LEVEL-2	21
5.3	LEVEL-3	22
5.4	LEVEL-4	22
5.5	USE CASE DIAGRAM	24
5.6	SEQUENCE DIAGRAM	25

CHAPTER-1

INTRODUCTION

The dataset was taken from Kaggle, comprised 7 CSV files containing data from 2009 to 2015, and was about 7GB in size. It contained information about airlines, delay information, location details (origins and destinations), and cancellation (reasons labelled as cancellation codes). It also had certain technical details such as the time that the plane was stationary on ground.

My approach revolved around Apache Spark , specifically using PySpark with AWS EMR cluster support. This virtual set-up uses distributed, in-memory data structures to improve speeds for many data processing workloads by several orders of magnitude. As a unified analytics engine, it allows us to parallelize large data processing tasks on a distributed cluster.

PySpark is also great for exploratory data analysis at scale, building machine learning pipelines, and creating ETLs (extract, transform, load) tools. Once the EMR cluster was “running,” I connected to Jupyterlab available as notebooks under AWS services and loaded the data from S3 buckets by importing it through the Kaggle website. I performed exploratory data analysis and model prediction.

1.1 EXISTING SYSTEM

In Existing system, an airport has huge amount of data related to number of flights, data and time of arrival and dispatch, flight routes, number of airports operating in each country, list of active airlines in each country. The problem they faced till now it's, they have ability to analyze limited data from databases.

1.2 PROPOSED SYSTEM

We want to analyze data regarding domestic flights to predict flight cancellations in the US. We will analyze the cancellation including delay time, number of passengers, airline carrier, scheduled and actual departure and arrival times, and reason

of delay. Using these values we plan to future calculate delays caused by the airport, get the count of the flights delayed with respect to the total number of flights etc.

We also aim to calculate the average airline delay time, which airline has the most number of delays, at what time of the day do we see more number of delays in flights. We also want to see if there are any seasonal delays i.e if delays are mostly during the holiday season. We will use Spark's MLlib to create powerful machine learning models alongside the DataFrame syntax and spark.

1.3 LITERATURE SURVEY

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy and company strength. Once these things are satisfied, then the next step is to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration are taken into account for developing the proposed system.

The major part of the project development sector considers and fully survey all the required needs for developing the project. For every project Literature survey is the most important sector in software development process. Before developing the tools and the associated designing it is necessary to determine and survey the time factor, resource requirement, man power, economy, and company strength. Once these things are satisfied and fully surveyed, then the next step is to determine about the software specifications in the respective system such as what type of operating system the project would require, and what are all the necessary software are needed to proceed with the next step such as developing the tools, and the associated operations.

CHAPTER 2

AIM AND SCOPE

2.1 Data analytics development of FDR (Flight Data Recorder) data for airline maintenance operations

In this article, we propose a data analytics development to detect unusual patterns of flights from a vast amounts of FDR (flight data recorder) data for supporting airline maintenance operations. A fundamental rationale behind this development is that if there are potential issues on mechanical parts of an aircraft during a flight, evidences for these issues are most likely included in the FDR data. Therefore, the data analysis of FDR data enables us to detect the potential issues in the aircraft before they occur. To this end, in a data pre-processing step, a data filtering, a data sampling, and a data transformation are sequentially performed. And then, in this analysis, all time series data in the FDR are classified into three types they are, a continuous signal, a discrete signal, and a warning signal.

For each type of signal, a high-dimensional vector by arranging the time series data is chosen as features. In the feature section process, a correlation analysis, a correlation relaxation, and a dimension reduction are sequentially conducted. Finally, a type of k-nearest neighbor approach is applied to automatically identify the FDR data in which the unusual flight patterns are recorded from a large amount of FDR data. The proposed method is tested with using a realistic FDR data from the NASA's open database.

2.2 Big Data Analytics on Aviation Social Media: The Case of China Southern Airlines on Sina Weibo

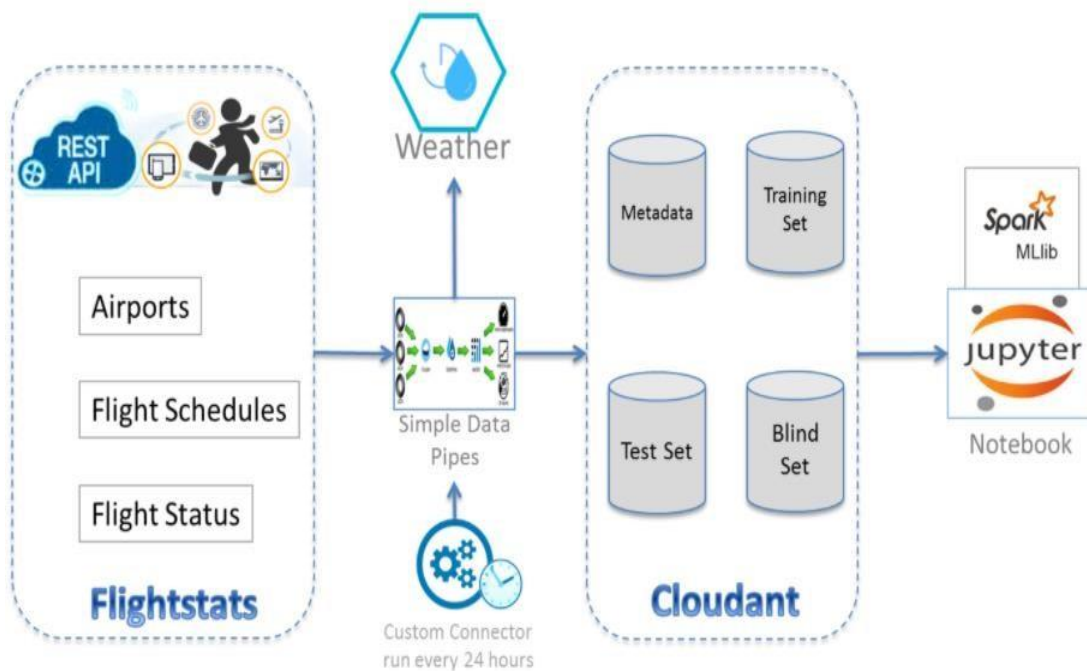
The proposed model using sentiment analysis to study the case of China Southern Airlines on Sina Weibo, and illustrating Weibo users' attitudes toward China Southern Airlines.

This study also discusses the practical implications for airlines to manage their social media platforms. With the combination of a passenger's social media value and

passenger's other offline behavior information, we can establish a comprehensive profile for travelers.

2.3 Big Data Analytics in Airlines: Efficiency Evaluation using DEA

This study aims to quantitatively evaluate the operational efficiency in the airline scheduling and execution process by implementing big data analytics approach. Parameters for calculation are obtained from prior studies.



2.1 ARCHITECTURE DIAGRAM

2.4 HARDWARE REQUIREMENTS

- System : Pentium Dual Core.
- Hard Disk : 120 GB.
- Monitor : 15" LED
- Input Devices : Keyboard, Mouse
- Ram : 4 GB.

2.5 SOFTWARE REQUIREMENTS

- Operating system: Windows 7/10.
- Coding Language: Python

CHAPTER 3

METHODOLOGY

3.1 SOFTWARE ENVIRONMENT

3.1.1 Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

3.1.2 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

3.1.3 Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below –

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

3.1.4 Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python <https://www.python.org>.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to <https://www.python.org/downloads/>.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

3.2 First Python Program

Let us execute programs in different modes of programming.

3.2.1 Interactive Mode Programming

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

```
$ python
Python2.4.3(#1,Nov112010,13:34:43)
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Type the following text at the Python prompt and press the Enter –

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result –

```
Hello, Python!
```

3.2.1 Script Mode Programming

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file –

```
print"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows –

```
$ python test.py
```

This produces the following result –

```
Hello, Python!
```

3.3 Flask Framework

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is part of the categories of the micro-framework. Micro-frameworks are normally framework with little to no dependencies to external libraries. This has pros and cons.

The following table summarizes different http methods –

Sr.No	Methods & Description
1	GET Sends data in unencrypted form to the server. Most common method.
2	HEAD Same as GET, but without response body
3	POST Used to send HTML form data to server. Data received by POST method is not cached by server.
4	PUT Replaces all current representations of the target resource with the uploaded content.
5	DELETE Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

```
<html>  
  
<body>
```

```

<form action="http://localhost:5000/login" method="post">
<p>Enter Name:</p>
<p><input type="text" name="nm"/></p>
<p><input type="submit" value="submit"/></p>
</form>
</body>
</html>

```

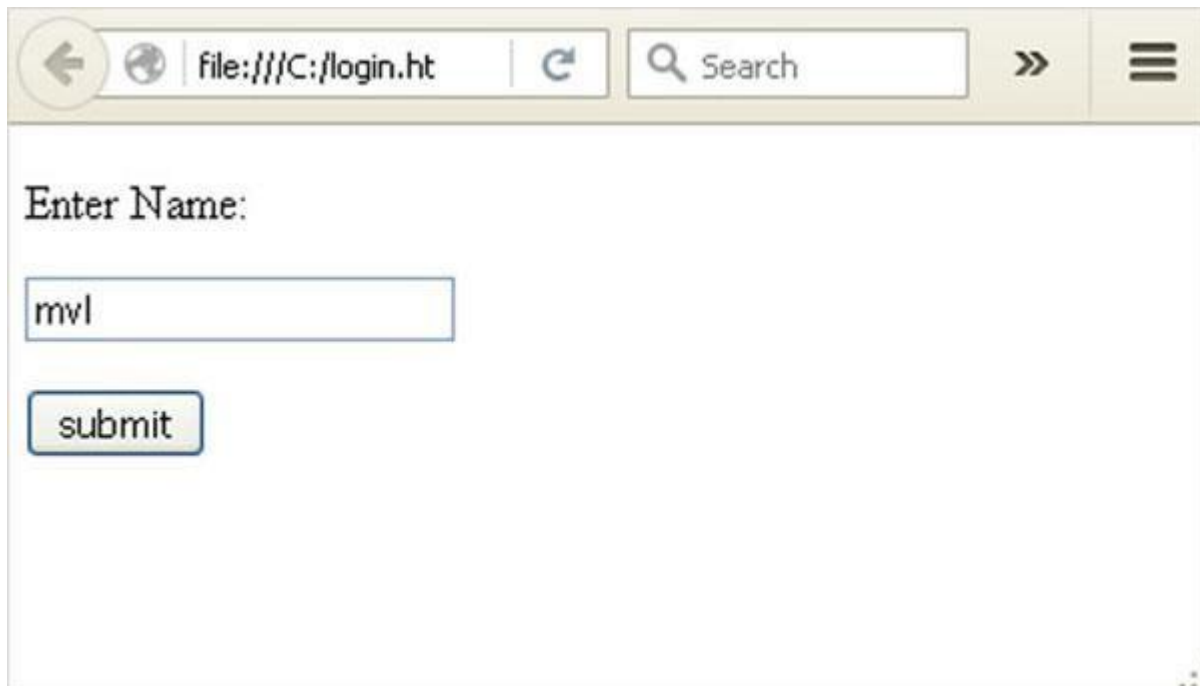
Now enter the following script in Python shell.

```

from flask import Flask, redirect, url_for, request
app = Flask(__name__)
@app.route('/success/<name>')
def success(name):
    return 'welcome %s' % name
@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        user = request.form['nm']
        return redirect(url_for('success', name= user))
    else:
        user = request.args.get('nm')
        return redirect(url_for('success', name= user))
if __name__ == '__main__':
    app.run(debug = True)

```

After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

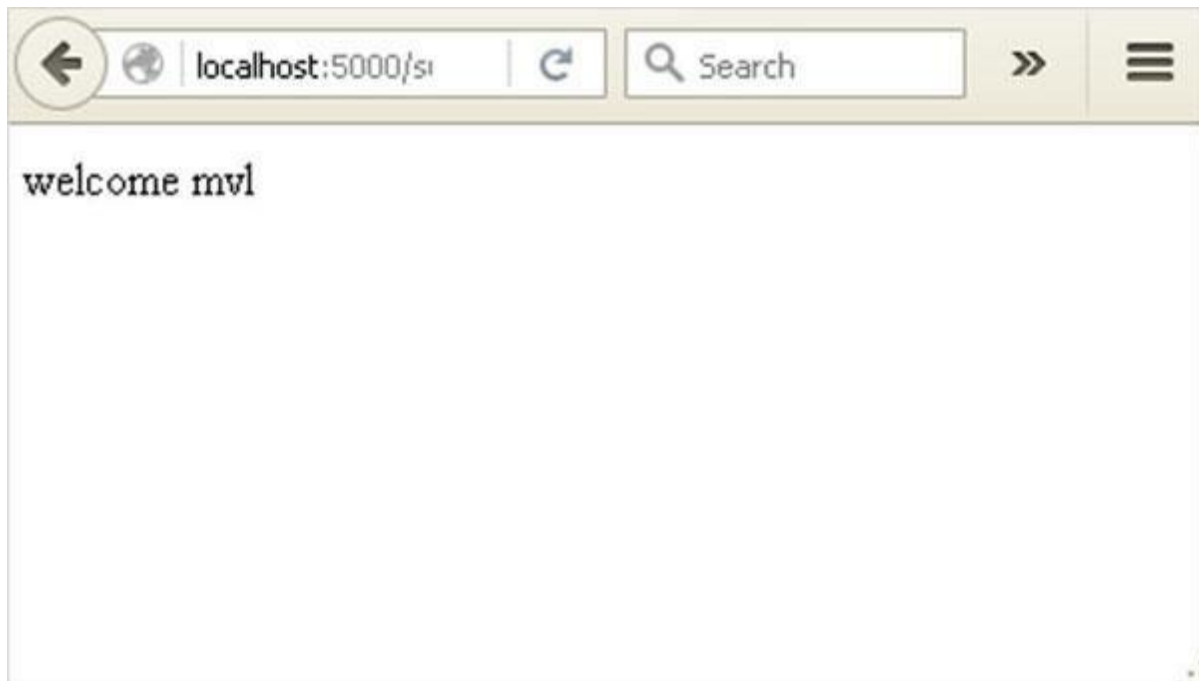
A screenshot of a web browser window. The address bar shows the file path 'file:///C:/login.ht'. The page content includes the text 'Enter Name:', a text input field containing 'mvl', and a 'submit' button.

Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of 'nm' parameter obtained from the form data is obtained by –

```
user = request.form['nm']
```

It is passed to **/success** URL as variable part. The browser displays a **welcome** message in the window.



Change the method parameter to **'GET'** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of 'nm' parameter is now obtained by –

```
User = request.args.get('nm')
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to 'nm' parameter is passed on to '/success' URL as before.

3.4 Modules

- Data Profiling
- Data Wrangling
- Data Cleaning and Preparation
- Statistical Data Analysis
- Model Prediction
- Random Forest Model

3.5 Data Profiling

To get a quick glance at the data, we loaded the 2009 csv file into a Resilient Distributed Dataset (RDD) which is Spark's representation of a dataset that is distributed across RAM, in the memory of a cluster of many machines. Through this Spark session, I found that there was about 27 unnamed variables and multiple null values. After combining the dataset (2009 to 2015 data), I created 3 major categorical CSV files namely: flight-cancellation, flight-delays, and flight-profile. This was essential to debug.

3.6 Data Cleaning and Preparation

Originally, there was 28 variables to work with. After removing the unnamed columns, the remaining 19 were checked for null values (as shown below). In addition, only columns with relevant data about the flight profile, delay, and cancellation information were kept. With time sensitive data, it was hard to recreate the null values with either mean or median of the data, so I had to drop these records which boils down to over 6+ million records to still work with. After cleaning out the date column, it contained only the yearly information which was used in the analysis.

CHAPTER 4

RESULT AND ANALYSIS

4.1 Statistical Data Analysis

- From the profile sub-dataset, there were 61,556,964 flight profile records including 7605 US domestic flights in total, around 380 unique origins and 378 unique destinations. This information was populated in temporary tables inside the Spark session to query specifics.
- From the delay sub-dataset, there were negative values in the departure delay column indicating that around 3,1204,918 flights were actually ahead of their scheduled time; that is roughly 50% of the dataset. This means that 50% of the time flights were delayed.
- Within the cancellation sub-dataset, the cancel column is categorical data represented by either a 0 or a 1 (not cancelled or cancelled respectively).

4.2 Model Prediction

To predict whether the flight would be cancelled or not, the dataset was modeled as a binary classification problem with categorical variables.

1. Prepare the data for machine learning: use StringIndexer, OneHotEncoder, and VectorAssembler to transform our features
2. Split the data into a 70/30 test and train ratio
3. Apply models: logistic regression, decision tree classifier, random forest and gradient boosted trees
4. Compare all the models' accuracy for predicting cancellations.

4.3 Decision Tree Classifiers

Decision Tree Classifiers can handle categorical variables well and are able to capture non-linearity. With the aid of the `pyspark.ml.classification` library, the `DecisionTreeClassifier` was imported. This was essential to show the probability of each prediction and evaluate the model.

4.4 Requirement analysis

Requirement analysis, also called requirement engineering, is the process of determining user expectations for a new modified product. It encompasses the tasks that determine the need for analysing, documenting, validating and managing software or system requirements. The requirements should be documentable, actionable, measurable, testable and traceable related to identified business needs or opportunities and define to a level of detail, sufficient for system design.

4.4.1 Functional Requirements

It is a technical specification requirement for the software products. It is the first step in the requirement analysis process which lists the requirements of particular software systems including functional, performance and security requirements.

4.4.2 Usability

It specifies how easy the system must be use. It is easy to ask queries in any format which is short or long, porter stemming algorithm stimulates the desired response for user.

4.4.3 Robustness

It refers to a program that performs well not only under ordinary conditions but also under unusual conditions. It is the ability of the user to cope with errors for irrelevant queries during execution.

4.4.4 Security

The state of providing protected access to resource is security. The system provides good security and unauthorized users cannot access the system there by providing high security.

4.4.5 Reliability

It is the probability of how often the software fails. The measurement is often expressed in MTBF (Mean Time Between Failures). The requirement is needed in order

to ensure that the processes work correctly and completely without being aborted. It can handle any load and survive and survive and even capable of working around any failure.

4.4.6 Compatibility

It is supported by version above all web browsers. Using any web servers like localhost makes the system real-time experience.

4.4.7 Flexibility

The flexibility of the project is provided in such a way that it has the ability to run on different environments being executed by different users.

4.4.8 Safety

Safety is a measure taken to prevent trouble. Every query is processed in a secured manner without letting others to know one's personal information.

4.5 System Design and Testing Plan

4.5.1 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

4.5.2 Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the

information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

CHAPTER 5

DESIGN AND TESTING

5.1 System Study

5.1.1 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

5.1.2 Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

5.1.3 Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed

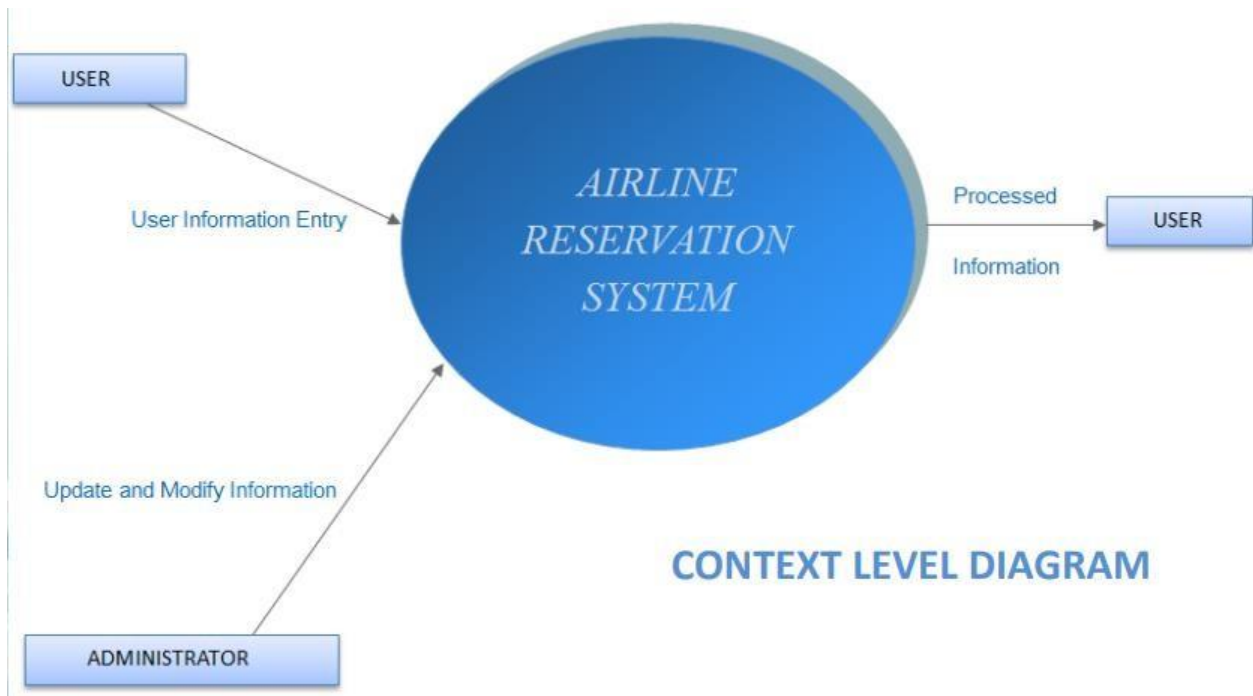
system must have a modest requirement, as only minimal or null changes are required for implementing this system.

5.1.4 Social Feasibility

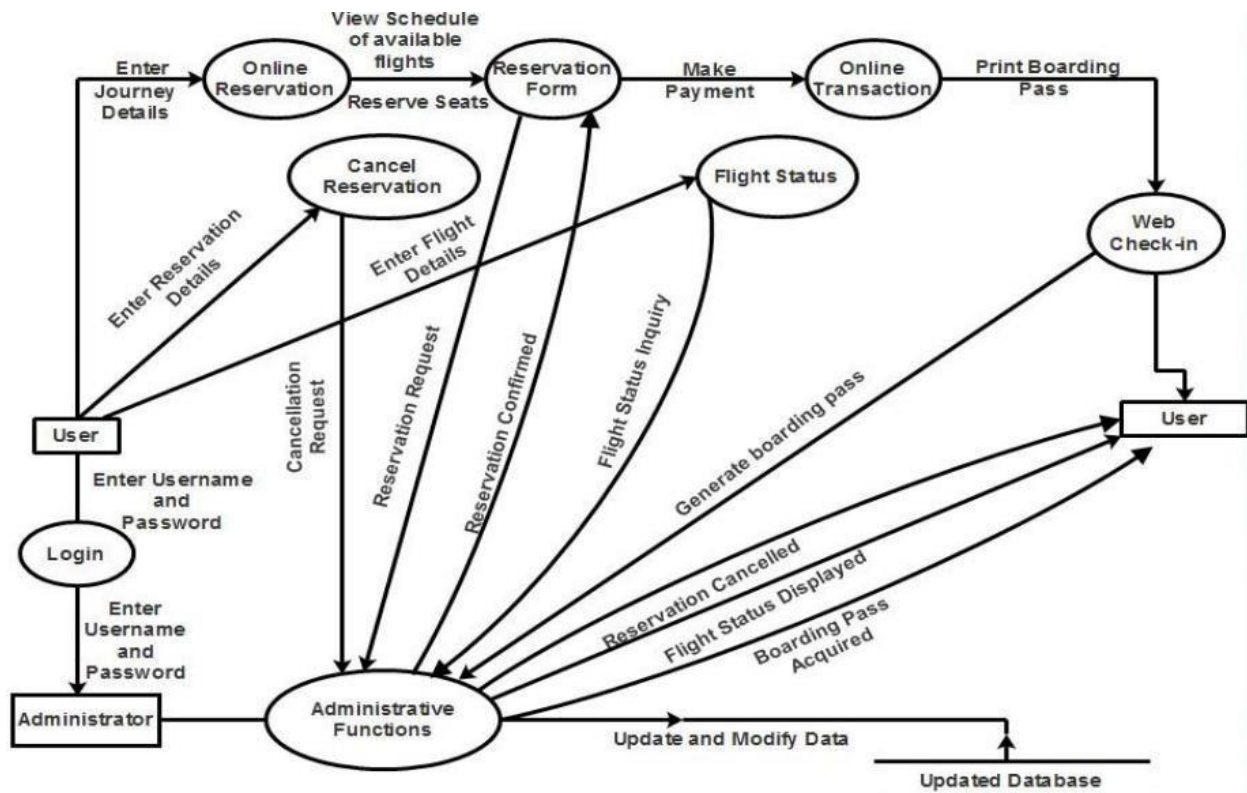
The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

5.2 Data Flow Diagrams

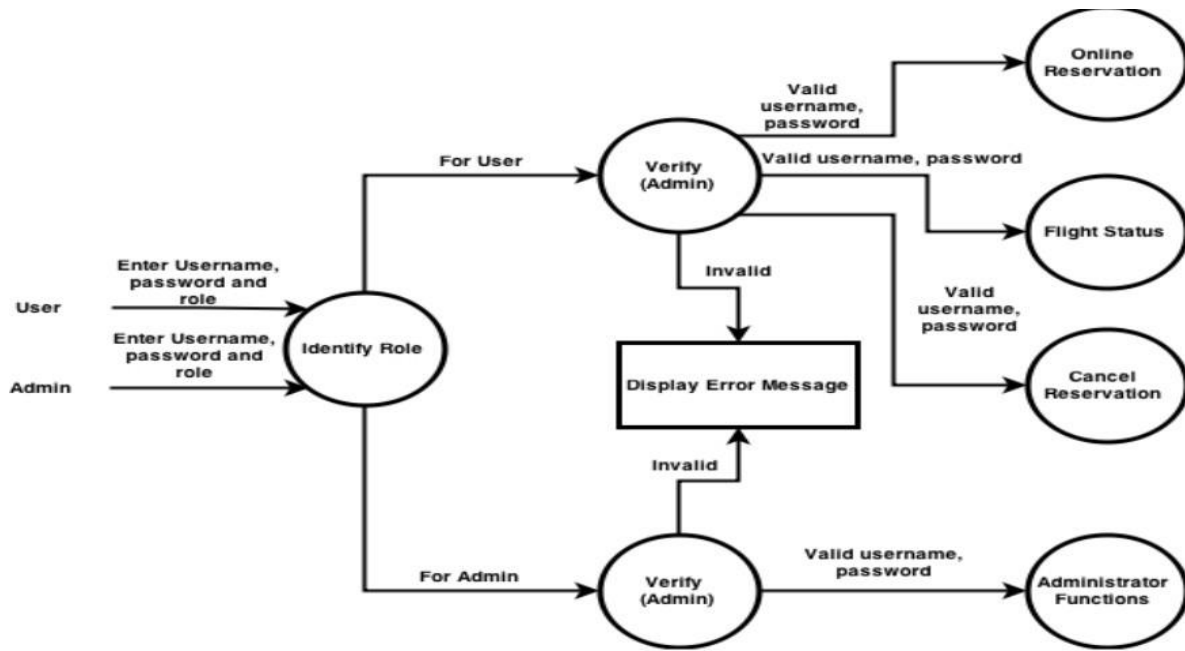
1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



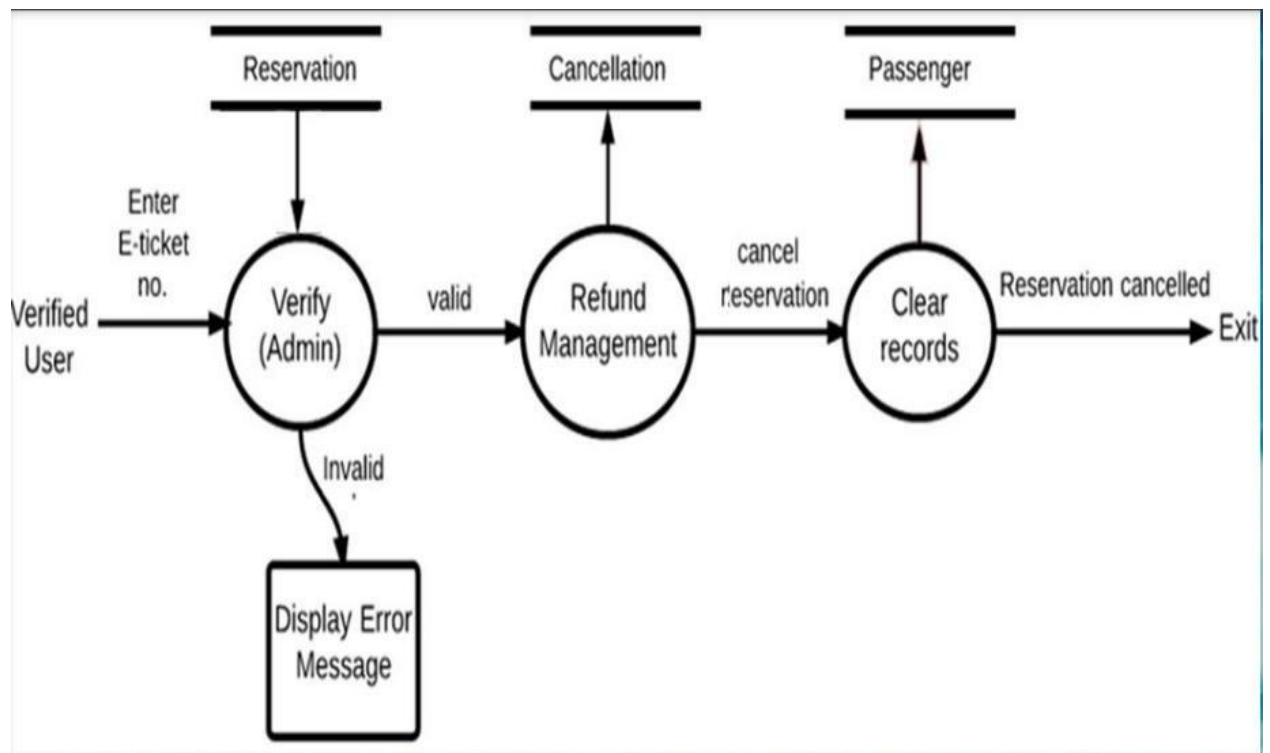
5.1 Context Level Diagram



5.2 Level-2



5.3 Level-3



5.4 Level-4

5.3 Uml Diagrams

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

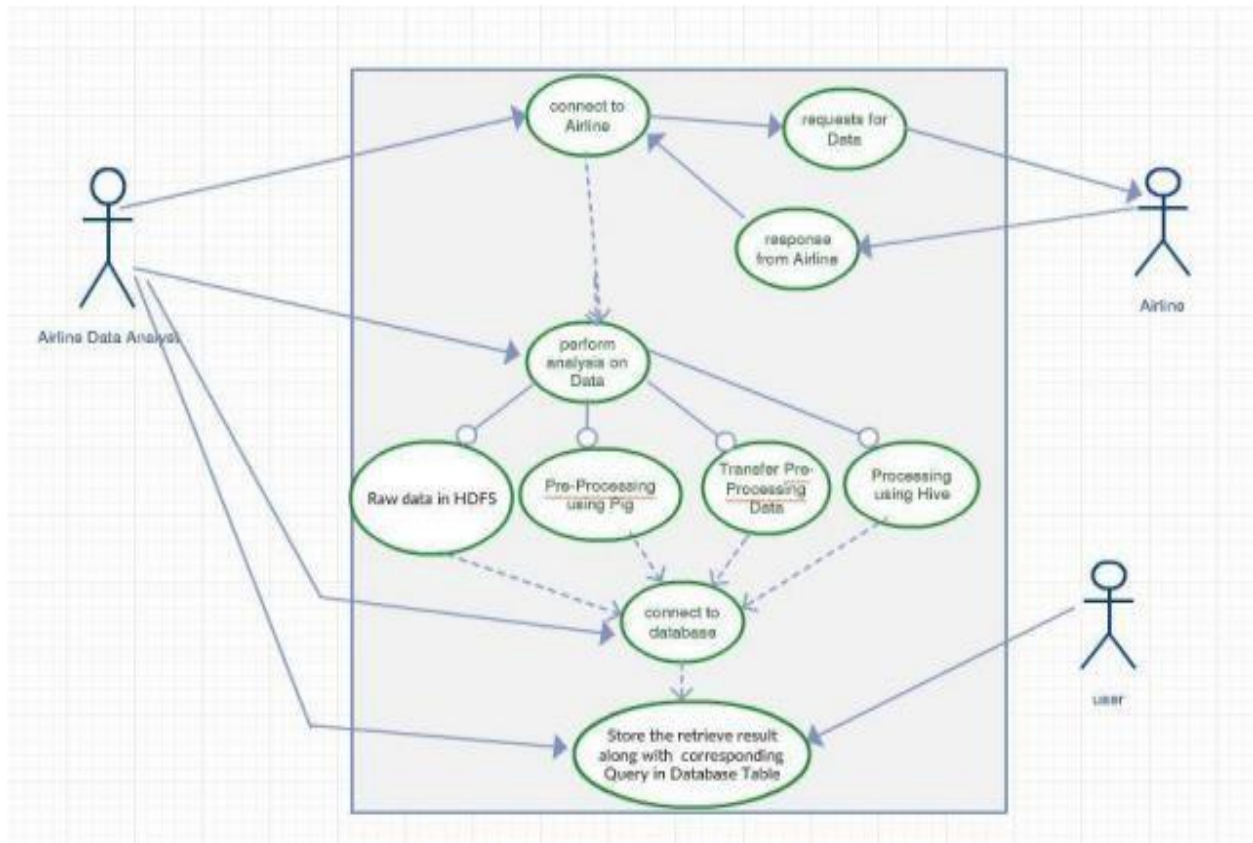
Goals

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

5.3.1 Use case Diagram

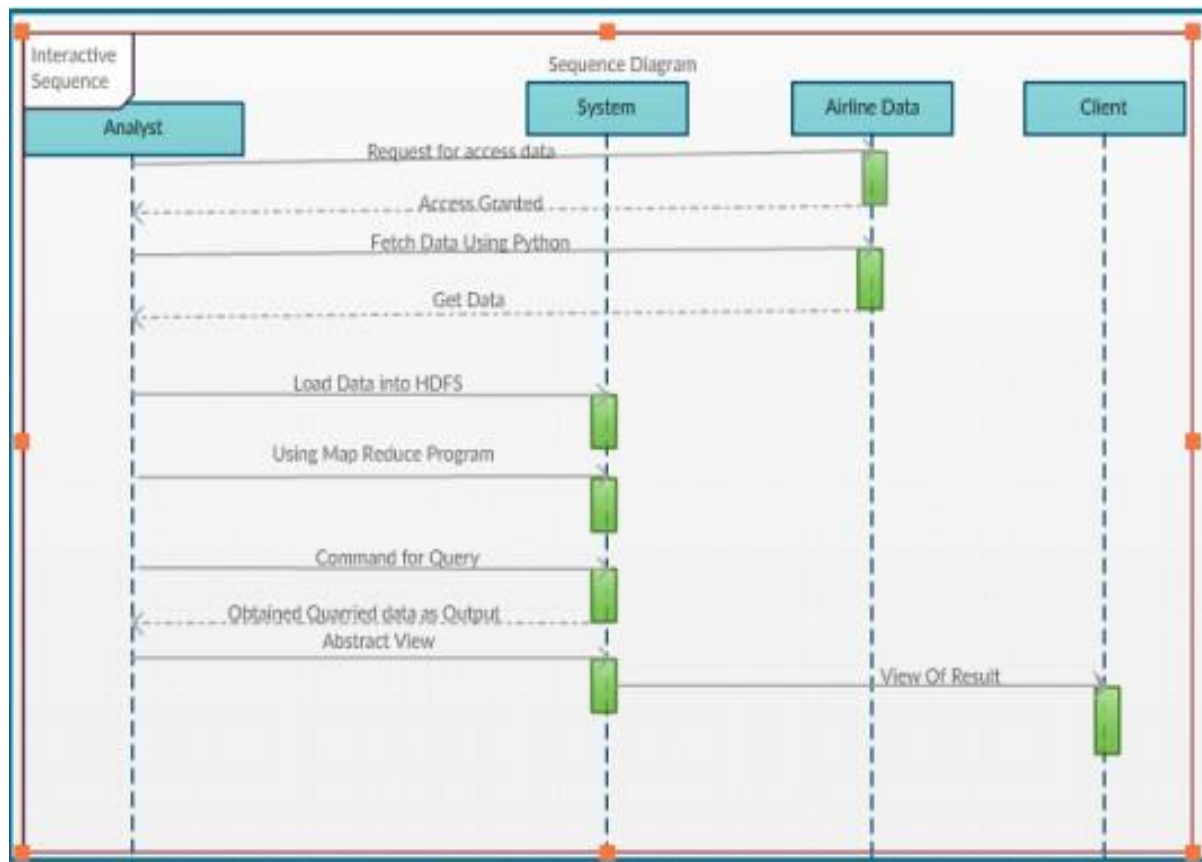
There are three actors in our project first one is the airline data analyst, second is airline and the third one is the user. The role of the analyst is to connect to the airline and then create an API which give the access to the to extract the data from airline. After getting access from airline using API .we can extract the airline data. Afterwards we will put the data into a excel table and insert it into HDFS after which the analysis one the particular topic. The analyst will receive the output by which the client will use the particular data.



5.5 Use case Diagram

5.3.2 Sequence Diagram

A Sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. There are four objects in our project which are airline data analyst, system interface, airline and client. first of all the process starts by the analyst by creating an API for airline data . the analyst request for the excess of airline data and then the access grated by the airline. Now the role of airline is done here, after we will program the raw data into HDFS and then insert into the excel table than it will show the extract view of the data. Now we are ready to fire the command and then we get the particular data as our output and provide to the clients.



5.6 Sequence Diagram

5.4 System Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements. In system testing, integration testing passed components are taken as input. ... It has both functional and non-functional testing.

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.4.1 Types of Tests

5.4.1.1 Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

5.4.1.2 Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

5.4.1.3 Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

5.4.1.4 System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5.4.1.5 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

5.4.1.6 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it.

5.4.1.7 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

5.5 Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

5.5.1 Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

5.5.2 Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

5.6 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or - one step up - software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

5.7 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 6

CONCLUSION

6.1 Conclusion

The project was able to identify whether the airline is likely to cancel with an accuracy of 63%. GBT performed the best for this case. From the data, the most likely flight to be cancelled was ExpressJet Airlines and the least likely was Delta Airlines.

However, airline identity was not a meaningful predictor of delay time. This suggests that delay of flight is likely to be for reasons other than the airline, such as weather conditions or airport. Ensuring security of the U.S. air travel system and of American and foreign air carriers traveling to and from the United States is a monumental task. Currently, in the United States, this involves screening over 1.5 million passengers and their carry-on baggage every day for the presence of metallic weapons or dangerous materials. The FAA is investigating and considering the implementation of new technologies that improve the effectiveness of current screening procedures by tailoring them to address and comply with current security requirements. At the same time, the FAA is looking into technologies that address anticipated changes in security requirements by expanding on the ability to detect the presence of dangerous materials on a person. After reviewing and assessing these efforts, the Panel on Passenger Screening reached the following conclusions.

REFERENCES

- [1] Bureau of Transportation Statistics. (2016). Airline On-Time Performance and Causes of Flight Delays. Retrieved from <https://catalog.data.gov/dataset/airline-on-time-performance-and-causes-of-flight-delays-on-time-data>
- [2] Deshpande, V., & Arikan, M. (2011). The Impact of Airline Flight Schedules on Flight Delays. *Manufacturing & Service Operations Management*, 14, 423-440. Retrieved from <https://pubsonline.informs.org/doi/10.1287/msom.1120.0379>
- [3] Mu, Y. (2019, August). Airline Delay and Cancellation Data, 2009 - 2018. Retrieved April 2020 from <https://www.kaggle.com/yuanyuwendymu/airline-delay-and-cancellation-data-2009-2018/data>
- [4] Chakrabarty, Navoneel, et al. "Flight Arrival Delay Prediction Using Gradient Boosting Classifier." *Emerging Technologies in Data Mining and Information Security*. Springer, Singapore, 2019. 651-659. Retrieved from https://www.researchgate.net/publication/327389509_Flight_Arrival_Delay_Prediction_Using_Gradient_Boosting_Classifier
- [5] Yi Ding "Predicting flight delay based on multiple linear regression", *IOP Conference Series: Earth and Environmental Science*. Retrieved from <https://iopscience.iop.org/article/10.1088/1755-1315/81/1/012198>
- [6] Belcastro, L. & Marozzo, Fabrizio & Talia, Domenico & Trunfio, Paolo. (2016). Using Scalable Data Mining for Predicting Flight Delays. *ACM Transactions on Intelligent Systems and Technology*. 8. 10.1145/2888402. Retrieved from <https://dl.acm.org/doi/10.1145/2888402>
- [7] Li, S. "Machine Learning with PySpark and MLlib – Solving a Binary Classification Problem". *Towards Data Science*. Retrieved from <https://towardsdatascience.com/machine-learning-with-pyspark-and-mllib-solving-a-binary-classification-problem-96396065d2aa>