Project Report

1. **INTRODUCTION**
   - 1.1 Project Overview
   - 1.2 Purpose
2. **LITERATURE SURVEY**
   - 2.1 Existing problem
   - 2.2 References
   - 2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
   - 3.1 Empathy Map Canvas
   - 3.2 Ideation & Brainstorming
   - 3.3 Proposed Solution
   - 3.4 Problem Solution fit
4. **REQUIREMENT ANALYSIS**
   - 4.1 Functional requirement
   - 4.2 Non-Functional requirements
5. **PROJECT DESIGN**
   - 5.1 Data Flow Diagrams
   - 5.2 Solution & Technical Architecture
   - 5.3 User Stories
6. **PROJECT PLANNING & SCHEDULING**
   - 6.1 Sprint Planning & Estimation
   - 6.2 Sprint Delivery Schedule
   - 6.3 Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
   - 7.1 Feature 1
   - 7.2 Feature 2
   - 7.3 Database Schema (if Applicable)
8. **TESTING**
   - 8.1 Test Cases
   - 8.2 User Acceptance Testing
9. **RESULTS**
   - 9.1 Performance Metrics
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**

    Source Code

    GitHub & Project Demo Link

1. INTRODUCTION

1.1 Project Overview:

Over the last twenty years, air travel has been increasingly preferred among travelers, mainly because of its speed and in some cases comfort. This has led to phenomenal growth in air traffic and on the ground. An increase in air traffic growth has also resulted in massive levels of aircraft delays on the ground and in the air. These delays are responsible for large economic and environmental losses. The main objective of the model is to predict flight delays accurately in order to optimize flight operations and minimize delays.

1.2 Purpose:

Using a machine learning model, we can predict flight arrival delays. The input to our algorithm is rows of feature vector like departure date, departure delay, distance between the two airports, scheduled arrival time etc. We then use decision tree classifier to predict if the flight arrival will be delayed or not. A flight is considered to be delayed when difference between scheduled and actual arrival times is greater than 15 minutes. Furthermore, we compare decision tree classifier with logistic regression and a simple neural network for various figures of merit.

2. LITERATURE SURVEY

2.1 Existing Problem:

Nowadays, the demand for airline transportation is increasing significantly. Analysis of flight delay, therefore, has become a popular research area. Various researchers used different techniques of machine learning and data mining to conduct the investigation. They were interested in different aspects such as airport facility location, weather condition, and airport capacity. Using machine learning allows researchers to handle large quantities of flight data for storing and processing.

Machine learning is the designation of algorithms that enable the computer to analyze the data, obtain potential patterns, and then use them to predict. Learning algorithms can give insight into the relative difficulty of learning in different environments . Machine learning algorithms are divided into several categories, and the two most common types are supervised learning and unsupervised learning. Algorithms of supervised learning generated a function that translates inputs to desired outputs. The primary forms of supervised learning algorithms include regression and classification. Unsupervised learning models a collection of inputs in the absence of labeled examples.

In this study, classification models were selected and trained using several algorithms: Logistic Regression, K-Nearest Neighbor (KNN), Gaussian Naïve Bayes, Decision

Tree, Support Vector Machine (SVM), Random Forest, and Gradient Boosted Tree. The first five of these algorithms are called base classifiers because only one classifier instance is trained for each one. The rest two of the algorithms are called ensemble classifiers because more than one instance of base classifiers are trained, and their collective decision is reported as the final prediction . As two of the most popular ensemble algorithms, Random Forest and Gradient Boosted Tree, combine several individual models to improve the performance by more accuracy and less variance.

References:

1) Waqar Ahmed Khan,Hoi-Lam Ma,Sai-Ho Chung,Xin Wen “Hierarchical integrated machine learning model for predicting flight departure delays and duration in series”Transportation Research Part C ,Elsevier,14 April 2021.

2) Qiang Li, Ranzhe Jing “Flight delay prediction from spatial and temporal perspective”,Expert Systems With Applications,Elsevier,8 May 2022

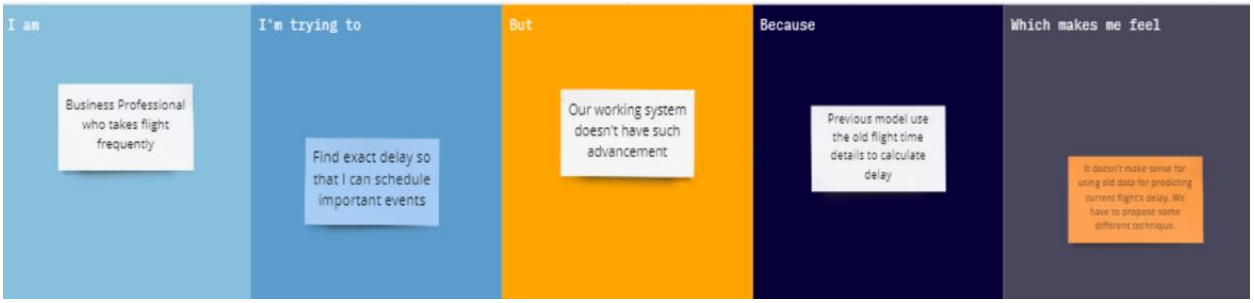
3) Zhen Guo a, Bin Yu a,b, Mengyan Hao a, Wensi Wang a, Yu Jiang c,Fang Zong, “A novel hybrid method for flight departure delay prediction using Random Forest Regression and Maximal Information Coefficient”,Aerospace Science and Technology,Elsevier,28 May 2021.

4) Seyedmirsajad Mokhtarimousavi,Armin Mehrabi, “Flight delay causality: Machine learning technique in conjunction with random parameter statistical analysis”,International Journal of Transportation Science and Technology,Elsevier,22 January 2022.

5) SP Lakshmi Narayanan1, S Sharbini2, R Priyanka3, R Kamalendran4, Mohit kumar5, C Murale, “FLIGHT DELAY PREDICTION USING SUPERVISED LEARNING”,International Research Journal of Engineering and Technology,Apr 2021

2.2Problem Statement Definition:

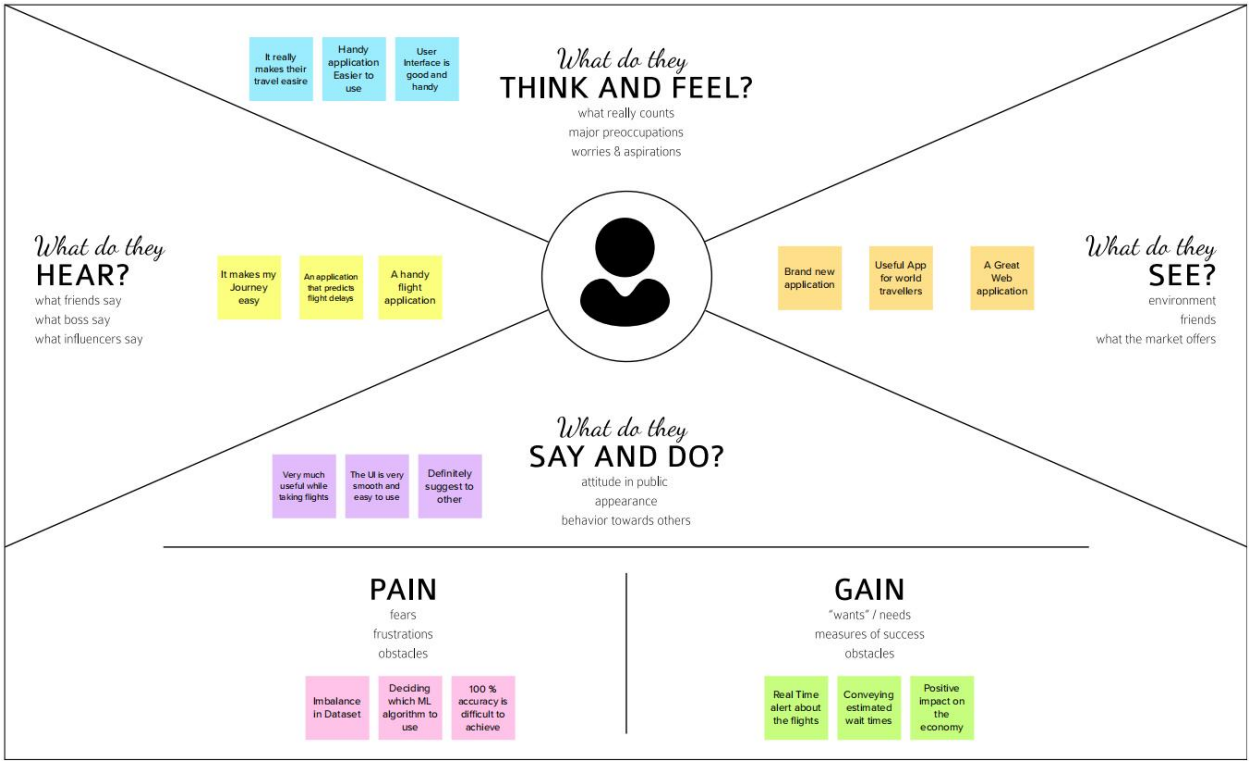
The main objective of the model is to predict flight delays accurately in order to optimize flight operations and minimize delays.



3. IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.



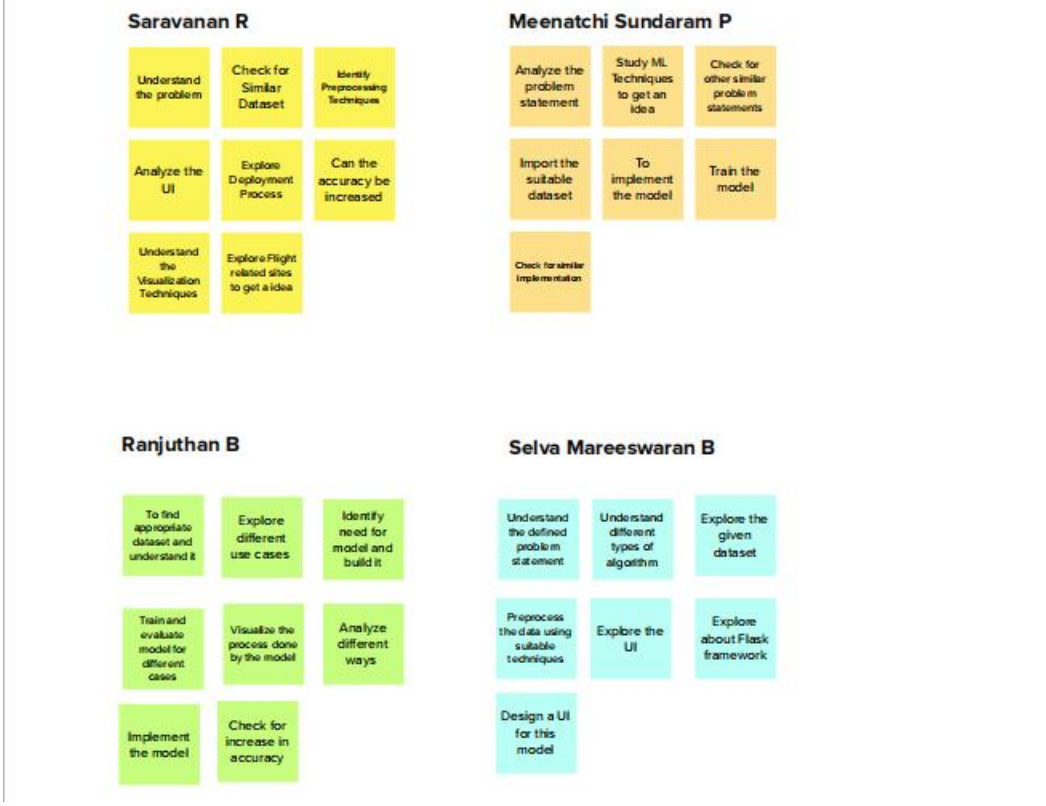
3.1 Ideation and Brainstorming:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon. All participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions. Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Brainstorm solo

Have each participant begin in the "solo brainstorm space" by silently brainstorming ideas and placing them into the template. This "silent-storming" avoids group-think and creates an inclusive environment for introverts and extroverts alike. Set a time limit. Encourage people to go for quantity.

⏱ 10 minutes



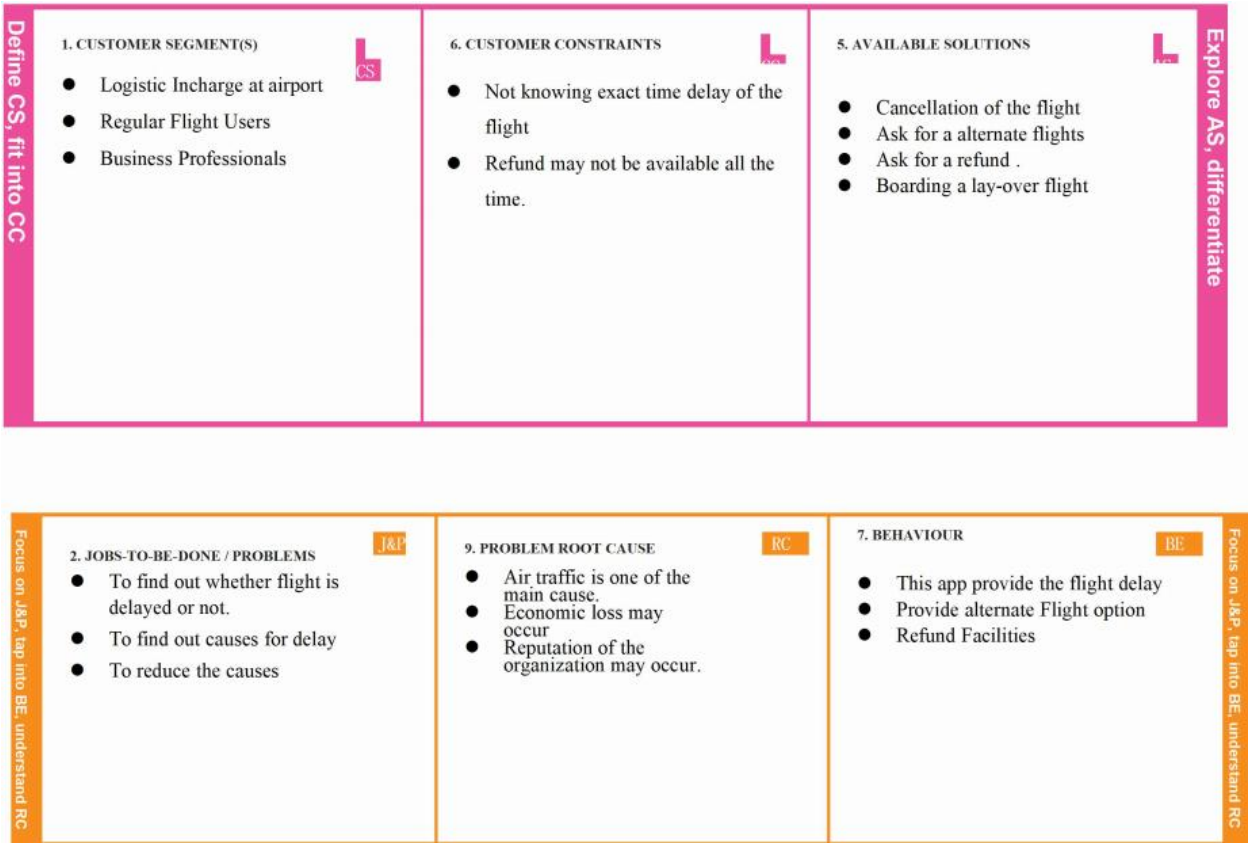
3.2Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Flight delays have been the most challenging area for airlines to improve. ● They have been affecting the air industry directly and indirectly causing unforeseen expenses thereby reducing the reputation of the industry and the airlines. ● Thus, knowing if a flight would be

		delayed beforehand can let passengers and airlines be prepared for the circumstances. ● This solution aims at making it possible by predicting arrival and departure delays using Machine learning
2.	Idea / Solution description	Building an application interface for customers(passengers and airlines) to know if a flight is delayed by implementing a machine learning based model to predict departure and arrival delays of an aircraft considering spatial, temporal and other dependencies causing the delay.
3.	Novelty / Uniqueness	● The solution takes into account all possible reasons for delay(crew delays, weather, air traffic, aircraft type) to provide an accurate prediction. ● Apart from predicting arrival delays, departure delays are also predicted in order for the passengers to prepare accordingly and for the airline to make arrangements suitably.
4.	Social Impact / Customer Satisfaction	A lot of time and money can be saved for the customers and the loyalty and trust of customers towards the company increases. Improves airline operations by letting the company prepare in prior to adversaries (like crew illness, timeouts, rescheduling) leading to passenger satisfaction which will result positively on the economy and brand value.
5.	Business Model (Revenue Model)	Business to Consumer model ● The solution is a low-cost airline model planned to be created as an application with which the consumers can interact directly to know the details of their flight. ● It follows a non-monetary revenue model where the consumers aren't charged for what they get but are asked to provide their flight details and ratings which can be used to improve the model and shared with the airline in return for airline's flight data
6.	Scalability of the Solution	● The present solution is drafted with the aim of experimenting with airlines based out of the United States of America. ● If there is a possibility to acquire data of a broader region (say North America, other

		continents), then the solution can be developed to benefit a wider range of people. ● International flight dependencies in both temporal and spatial focus can be derived from that data to provide more accurate predictions. ● Presence of ADS-B data can further increase the efficiency of system making it reach global audience and live time tracking of flights.

3.1 Problem Solution Fit:



4. Requirement Analysis

4.1 Functional Requirement:

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Login	Log in using given credentials
FR-4	Delay Prediction	Requesting for details from user for making the predictions about their flight
FR-5	Feedback	Get feedback from customer about their experience
FR-6	Logout	Logout from the session

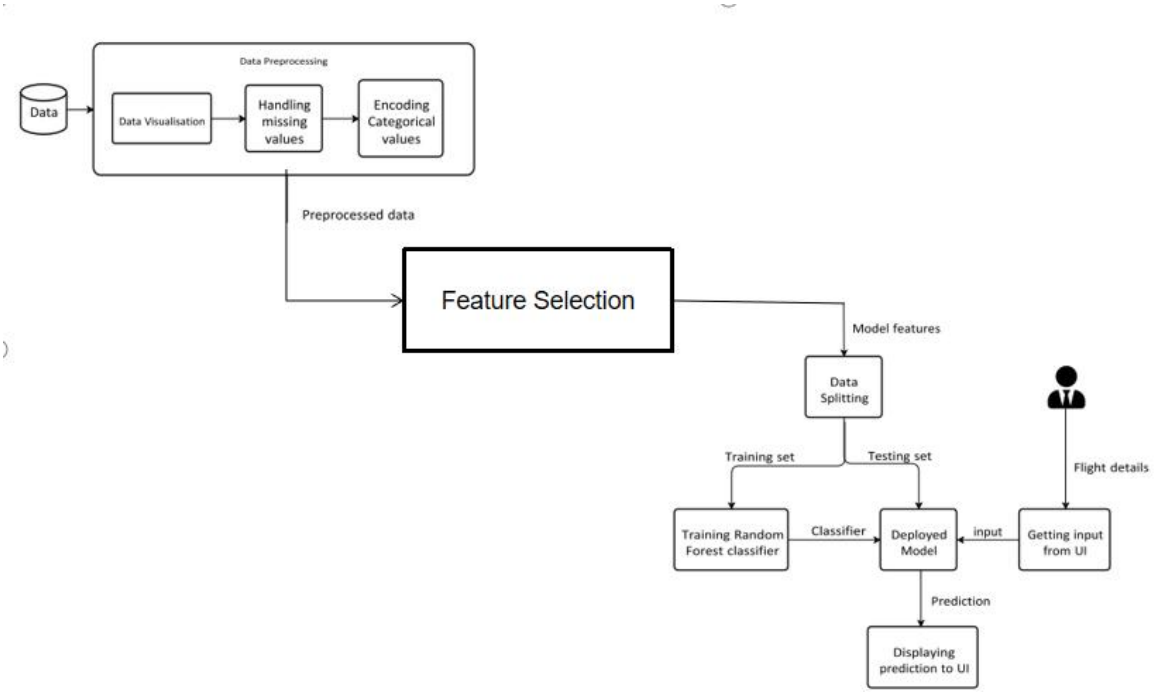
Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

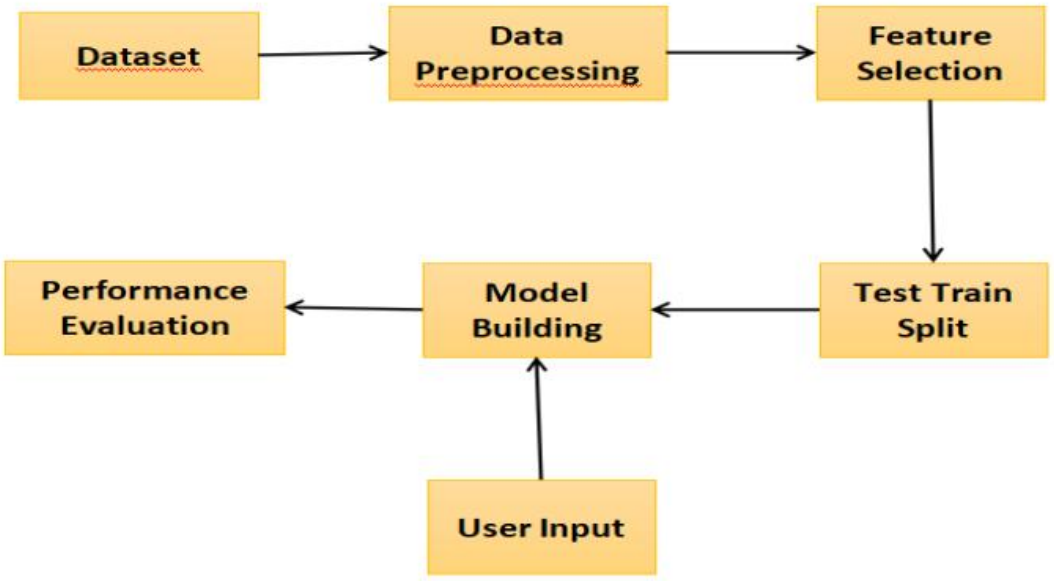
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The app must have such ease in usability. The UI should be simple and clear for the user to understand
NFR-2	**Security**	A 2 step verification should be implemented in the application All the confidential information about the flight and customer must be confidential
NFR-3	**Reliability**	The application must be able to work under any situation. It should be able to restore all its content even if the system fails
NFR-4	**Performance**	The application should perform in a faster way It shouldn't take much time for prediction of delay The time should be less than 3 seconds.
NFR-5	**Availability**	The application should be available 24/7 .

5. PROJECT DESIGN

5.1 Data Flow Diagrams:



5.2 Solution Architecture:



Technological Architecture:

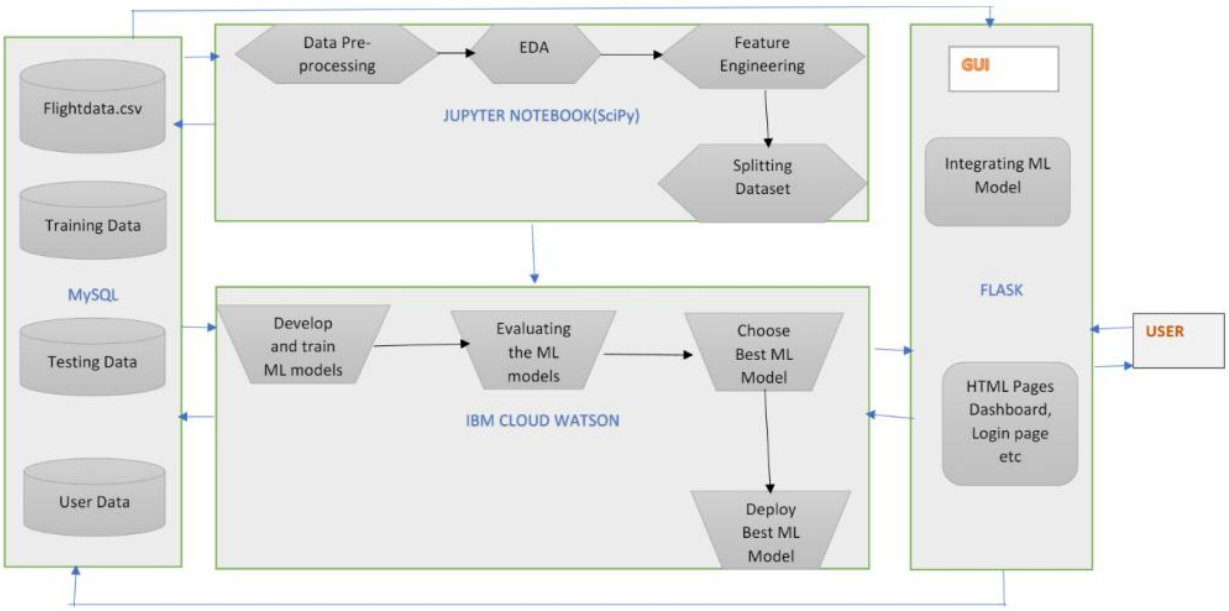


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Develop and train the mode;	Python
3.	Application Logic-2	Deploy the model in IBM cloud	IBM Watson STT service
4.	Application Logic-3	Integrating the python model with Flask environment	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL, NoSQL, etc.
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloudant etc.
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	External API-1	To deploy model in cloud	IBM Weather API, etc.
9.	Machine Learning Model	To perform the core concept	Object Recognition Model, etc.
10.	Infrastructure (Server / Cloud)	App deployment on cloud	Kubernetes

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Jupyter Notebook

S.No	Characteristics	Description	Technology
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	e.g. SHA-256, Encryptions, IAM Controls, OWASP etc.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	GUI
4.	Availability	Justify the availability of application (e.g. use of load balancers, distributed servers etc.)	IBM Cloud
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	IBM Watson App Service

User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-2
		USN-4	As a user, I can register for the application through Gmail	I can register successfully with gmail login	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can login using my email and password	High	Sprint-1
	Dashboard	USN-6	As a user, I can use the dashboard to enter the details of my flight like arrival time, flight id,destination,source place,etc…	I can enter the details into the textfield	High	Sprint-2
Customer (Web user)	Predict Delay	USN-7	As a user , I can use this button to calculate or predict the delay of my flight	I can push the button to get the result	High	Sprint-2
Customer Care Executive	Feedback Section	USN-8	As a user , I can give my feedback in this section	I am able to give my feedback.	Medium	Sprint-3
Administrator	Maintenance	USN-9	As an administrator,I can be able to monitor all the tasks going on and can modify any changes	I can have the control to all section to modify anything.	Medium	Sprint-4

6 PROJECT PLANNING AND SCHEDULING

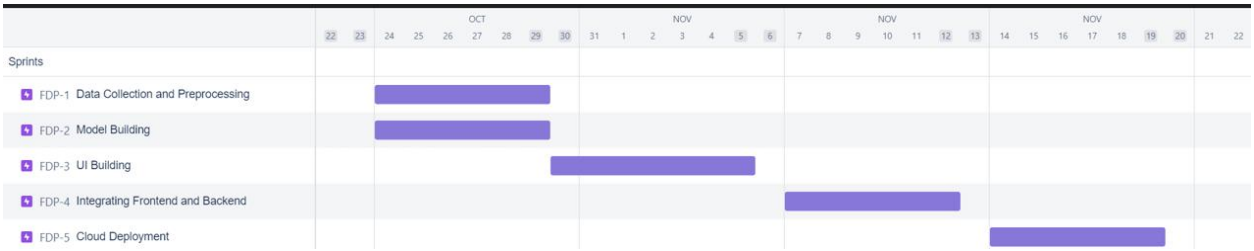
6.1Sprint Planning and Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Data Collection, Preprocessing	USN-1	Before working on the model we must have some data . So the first thing to do is to collect the data and then after data collection we have to preprocess it.	2	High	P Meenatchi Sundaram, B. Selva Mareeswaran
Sprint-1	Model Building	USN-2	After preprocessing we have to put the preprocessed data into ML models to know how to work .	1	High	R.Saravanan B.Ranjuthan
Sprint-2	UI	USN-3	The frontend has to be created . So that the data can be retrieved from the user.	2	Medium	B.Ranjuthan
Sprint-3	Integrating frontend and Backend	USN-4	After creating the frontend and backend both has to be integrated so that it will be a full fledged application	2	High	B.Ranjuthan P.Meenatchi Sundaram R.Saravanan
Sprint-4	Cloud Deployment	USN-5	After integrating we have to deploy the whole into cloud	1	High	B.Selva Mareeswaran P.Meenatchi Sundaram R.Saravanan B.Ranjuthan

6.2 Sprint Delivery Schedule

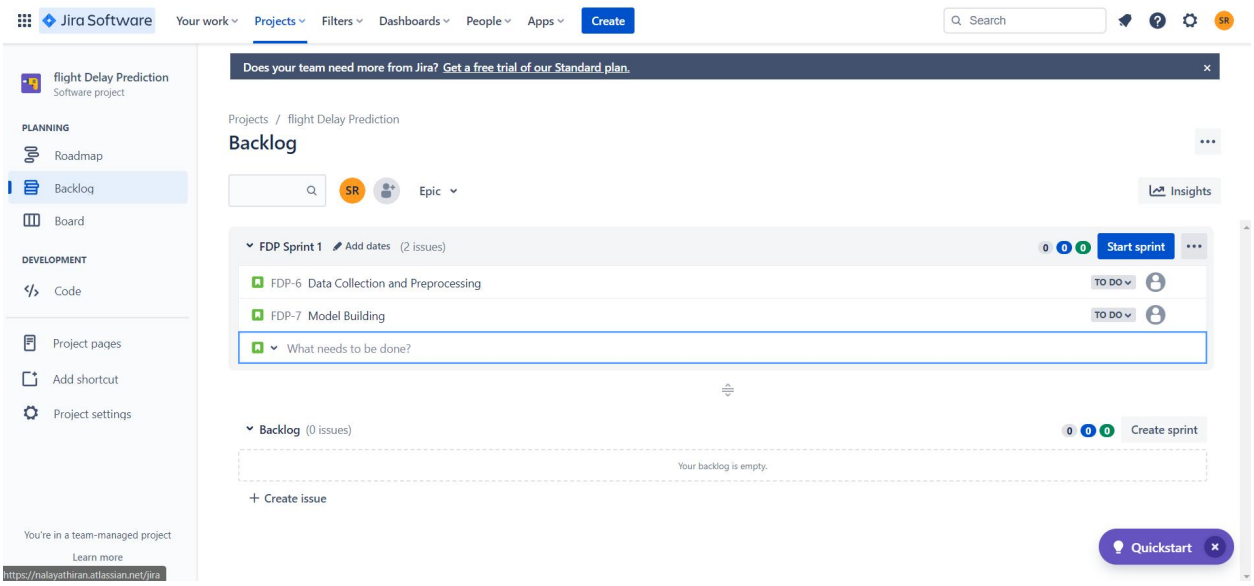
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	7 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Roadmap :

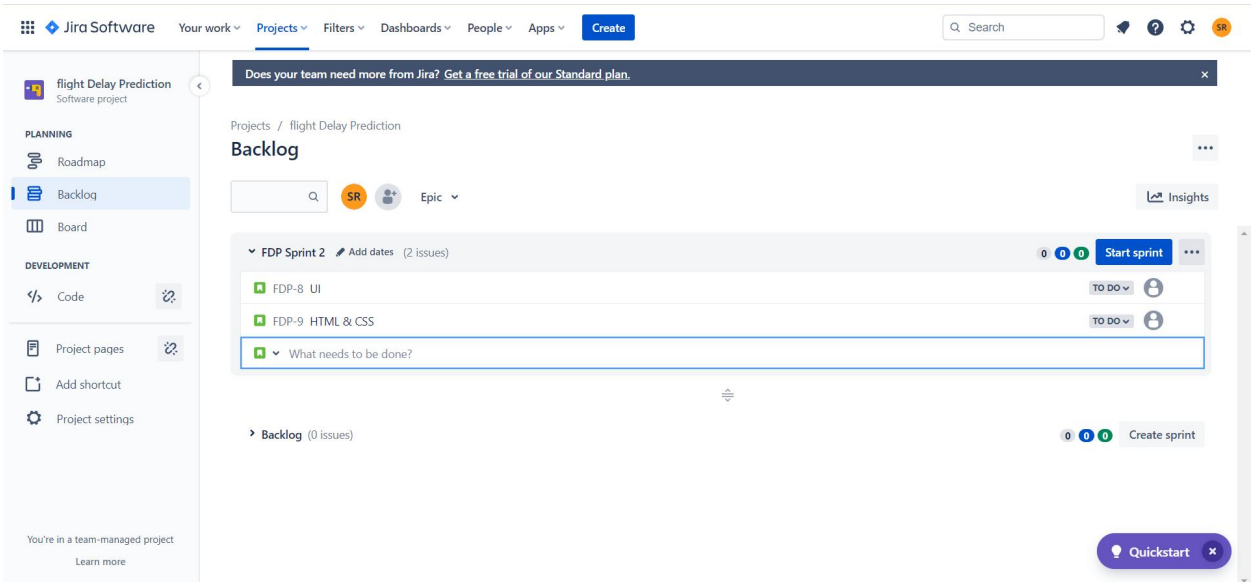


6.3 Reports from JIRA:

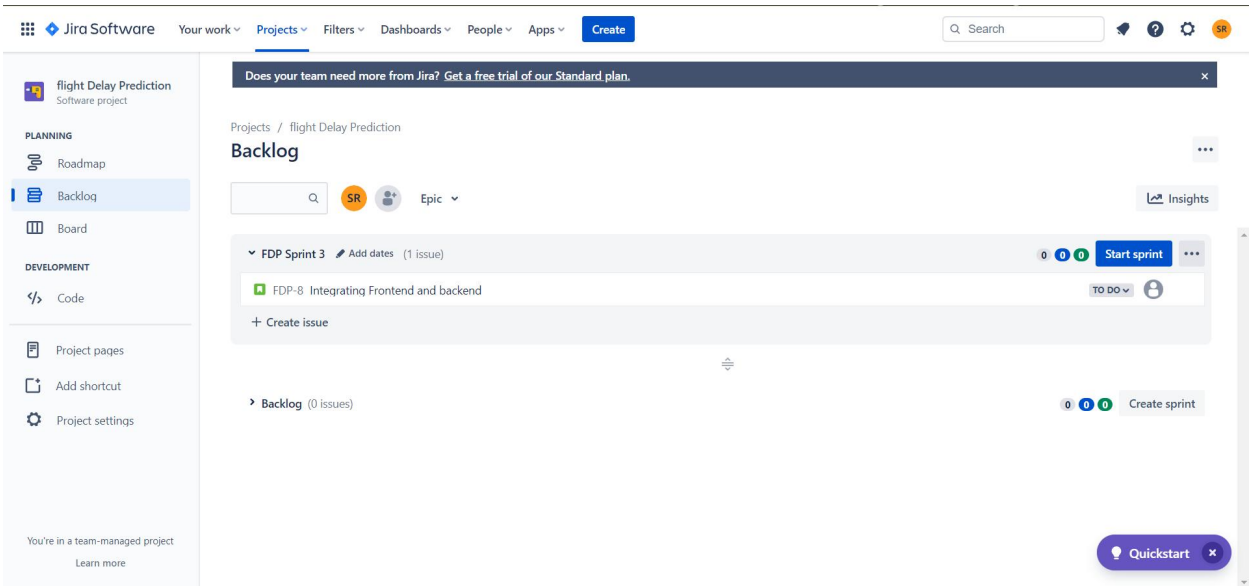
Sprint 1 :



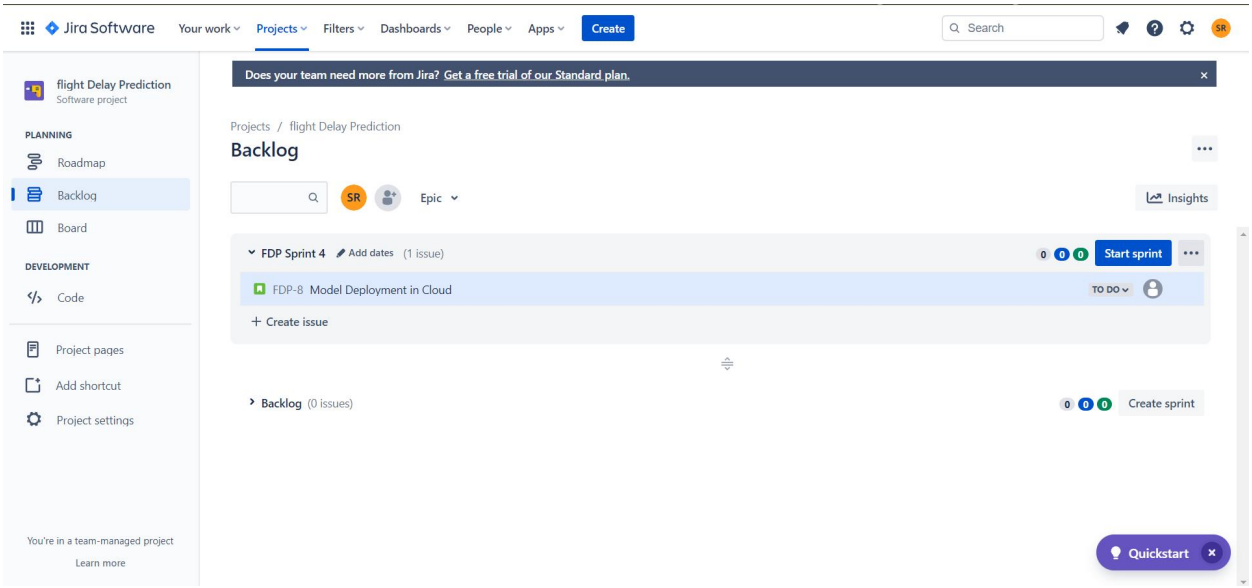
Sprint 2 :



Sprint 3:



Sprint 4:



7 CODING & SOLUTIONING

7.1 App.py

```
from flask import Flask, render_template, request
import pandas as pd
import joblib
import numpy as np

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('Flightdelay.html')

@app.route('/result', methods=['POST'])
def predict():
    fl_num = int(request.form.get('fno'))
    month = int(request.form.get('month'))
    dayofmonth = int(request.form.get('daym'))
    dayofweek = int(request.form.get('dayw'))
    origin = str(request.form.get("org"))
    dest = str(request.form.get("dest"))
    sdeptime = request.form.get('sdt')
    sarrtime = int(request.form.get('sat'))
    adeptime = request.form.get('adt')

    inputs = list()
    inputs.append(fl_num)
    inputs.append(month)
    inputs.append(dayofmonth)
```

```
    inputs.append(dayofweek)
    inputs.append(sdeptime)
    inputs.append(sarrtime)
    inputs.append(adeptime)

    if(origin == "ATL"):
        a = [1, 0, 0, 0, 0]
        inputs.extend(a)
    elif(origin == "DTW"):
        a = [0, 1, 0, 0, 0]
        inputs.extend(a)
    elif(origin == "JFK"):
        a = [0, 0, 1, 0, 0]
        inputs.extend(a)
    elif(origin == "MSP"):
        a = [0, 0, 0, 1, 0]
        inputs.extend(a)
    elif(origin == "SEA"):
        a = [0, 0, 0, 0, 1]
        inputs.extend(a)

    if(dest == "ATL"):
        b = [1, 0, 0, 0, 0]
        inputs.extend(b)
    elif(dest == "DTW"):
        b = [0, 1, 0, 0, 0]
        inputs.extend(b)
    elif(dest == "JFK"):
        b = [0, 0, 1, 0, 0]
        inputs.extend(b)
    elif(dest == "MSP"):
        b = [0, 0, 0, 1, 0]
        inputs.extend(b)
    elif(dest == "SEA"):
        b = [0, 0, 0, 0, 1]
        inputs.extend(b)

    # inputs.append(origin)
    # inputs.append(dest)

    print(inputs)
    prediction = preprocessAndPredict(inputs)
    # Pass prediction to prediction template
    return render_template('/result.html', prediction=prediction)

def preprocessAndPredict(inputs):
    test_data = np.array(inputs).reshape((1, 17))
    # test_data = inputs

    model_file = open('model_forest_reg.pkl', 'rb')
```

```
    trained_model = joblib.load(model_file)

    # df = pd.DataFrame(data=test_data[0:, 0:], columns=['FL_NUM', 'MONTH',
'DAY_OF_MONTH','DAY_OF_WEEK', 'origin', 'dest', 'CRS_DEP_TIME', 'CRS_ARR_TIME', 'DEP_TIME'])

    df = pd.DataFrame(data=test_data[0:, 0:], columns=['FL_NUM', 'MONTH', 'DAY_OF_MONTH',
'DAY_OF_WEEK', 'CRS_DEP_TIME', 'CRS_ARR_TIME', 'DEP_TIME',
                'ORIGIN_ATL', 'ORIGIN_DTW', 'ORIGIN_JFK', 'ORIGIN_MSP', 'ORIGIN_SEA', 'DEST_ATL',
'DEST_DTW', 'DEST_JFK', 'DEST_MSP', 'DEST_SEA'])

    data = df.values

    result = trained_model.predict(data)

    print(result)
    return result

if __name__ == '__main__':
    app.run(debug=True)
```

7.2 UI*:*

```
<!DOCTYPE html>

<html lang="en">

<head>

        <meta charset="UTF-8">

        <meta http-equiv="X-UA-Compatible">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" href="{{ url_for('static',filename='styles/styles.css') }}">

        <script src="{{url_for('static', filename='styles/delaypredict.js')}}"></script>

        <title>Flight Delay Prediction</title>

</head>

<body id="flight-form">

<h2 id="main-head" class="centered-head">FLIGHT DELAY PREDICTION</h2>

<img src="{{url_for('static', filename='styles/images/Flight1.jpg')}}" id="bgimg">

<form name="flightForm" action="/result" method="POST" target="_blank">
```

```
<div id="form-content">
    <div id="block1">
        <div class="detail-container">
            <label for="fno" class="label-item">Enter the Flight Number</label>
            <br>
            <input type="number" id="fno" name="fno" class="text-input">
        </div>
        <div class="detail-container">
            <label for="month" class="label-item">Month</label>
            <br>
            <input type="number" id="month" name="month" class="text-input" onblur="checkValid('month');" placeholder="Enter the Month Number">
            <div class="alert-text" id="month-valid">Enter a valid month between 1 to 12.</div>
        </div>
        <div class="detail-container">
            <label for="daym" class="label-item">Day of Month</label>
            <br>
            <input type="number" id="daym" name="daym" class="text-input" onblur="checkValid('daym');">
            <div class="alert-text" id="daym-valid">Enter a valid day of month.</div>
        </div>
        <div class="detail-container">
            <label for="dayw" class="label-item">Day of Week</label>
            <br>
            <input type="number" id="dayw" name="dayw" class="text-input" onblur="checkValid('dayw');">
```

```
                    <div class="alert-text" id="dayw-valid">Enter a valid day between 1
to 7.</div>
                </div>
                <div class="detail-container">
                    <label for="org" class="label-item">Origin</label>
                    <br>
                    <select id="org" name="org" class="select-input">
                        <option value="ATL" class="option-item">ATL</option>
                        <option value="SEA" class="option-item">SEA</option>
                        <option value="DTW" class="option-item">DTW</option>
                        <option value="MSP" class="option-item">MSP</option>
                        <option value="JFK" class="option-item">JFK</option>
                    </select>
                </div>
                <div class="detail-container">
                    <label for="dest" class="label-item">Destination</label>
                    <br>
                    <select id="dest" name="dest" class="select-input"
onblur="checkValid('dest');">
                        <option value="ATL" class="option-item">ATL</option>
                        <option value="SEA" class="option-item">SEA</option>
                        <option value="DTW" class="option-item">DTW</option>
                        <option value="MSP" class="option-item">MSP</option>
                        <option value="JFK" class="option-item">JFK</option>
                    </select>
                    <div class="alert-text" id="dest-valid">Enter different Origin and
Destination.</div>
```

```
            </div>
        </div>
        <div id="block2">
            <div class="detail-container">
                <label for="sdt" class="label-item">Scheduled Departure Time</label>
                <br>
                <input type="number" id="sdt" name="sdt" class="text-input" onblur="checkValid('sdt');" placeholder="Enter in the format HHMM">
                <div class="alert-text" id="sdt-valid">Enter a valid time between 500 to 2359.</div>
            </div>
            <div class="detail-container">
                <label for="sat" class="label-item">Scheduled Arrival Time</label>
                <br>
                <input type="number" id="sat" name="sat" class="text-input" onblur="checkValid('sat');" placeholder="Enter in the format HHMM">
                <div class="alert-text" id="sat-valid">Enter a valid time between 500 to 2359.</div>
            </div>
            <div class="detail-container">
                <label for="adt" class="label-item">Actual Departure Time</label>
                <br>
                <input type="number" id="adt" name="adt" class="text-input" onblur="checkValid('adt');" placeholder="Enter in the format HHMM">
                <div class="alert-text" id="adt-valid">Enter a valid time between 500 to 2359.</div>
            </div>
        </div>
```

```
        </div>

        <div id="submit-button">

                <input type="submit" value="Submit" id="submit" class="button"
onclick="validateForm()">

        </div>

</form>

</body>

</html>
```

8TESTING

8.1.1Test Cases:

1					Date	03-Nov-22				
2					Team ID	PNT2022TMID17945				
3					Project Name	Developing a Flight Delay Predictio				
4					Maximum Marks	4 marks				
5	**Test case ID**	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**
6	Dashboard	Functional	Home Page	User can enter the details about the flight like arrival , departure time , flight number	Any Browser	1.Enter URL and click go 2.Enter valid flight number 3.Enter Valid date in terms of month, day of month and day of week 4.Enter valid origin and destination 5.Enter valid flight timings 6.Click submit	Flight Number: 2610 Month: 1 Day of Month: 2 Day of Week: 6 Origin: ATL Destination: MSP Scheduled Dept Time: 1210 Actual Dept Time: 1231	User should be navigated to the next page like whether the flight is delayed or not	Working as expected	Pass
7	Dashboard2	UI	Output Page	Displays whether flight is delayed or not	Any Browser	In the first page click on predict it will redirect to this page	Flight Number: 2610 Month: 1 Day of Month: 2 Day of Week: 6 Origin: MSP Destination: MSP Scheduled Dept Time: 1210 Actual Dept Time: 1231 Scheduled Arr Time: 1415	Application should show "Your Flight is on time " or " Your Flight is delayed"	Working as expected	Pass
8	PredictionPage_TC_OO1	Functional	Home Page	Verify user is able to see the Prediction input page when user clicked on URL	Any Latest Browser	1.Enter URL and click go 2.Verify if the Prediction input is fully displayed or not	Flask App	The Prediction input page should be displayed	Working as expected	Pass
9	PredictionPage_TC_OO2	UI	Home Page	Verify the UI elements in Prediction page	Any Latest Browser	1.Enter URL and click go 2.Verify Prediction page with below UI elements: a.Flight number b.Date in terms of month, day of month and day of week c.Origin and Destination dropdown d.Flight Timings e.Submit button	Flask App	Application should show below UI elements: a.Flight Number b.Date in terms of month, day of month and day of week c.Origin and Destination dropdown d.Flight Timings e.Submit button	Working as expected	Pass

8.2User Acceptance Testing:

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Developing a Flight Delay Prediction Model using Machine Learning project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	40	0	0	40
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	3	0	0	3
Version Control	1	0	0	1

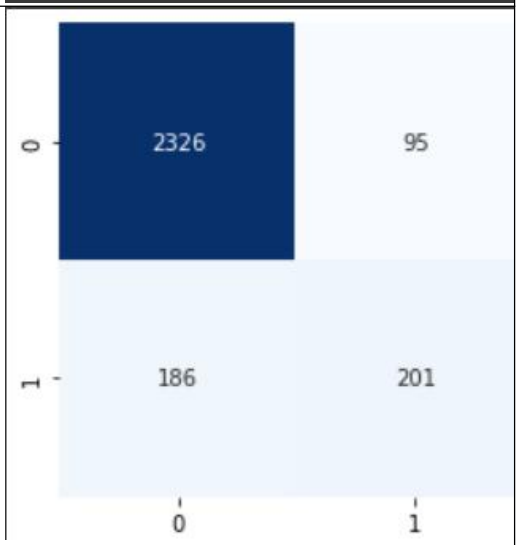
9 RESULTS:

Performance Metrics:

S .No.	Parameter	Values	Screenshot

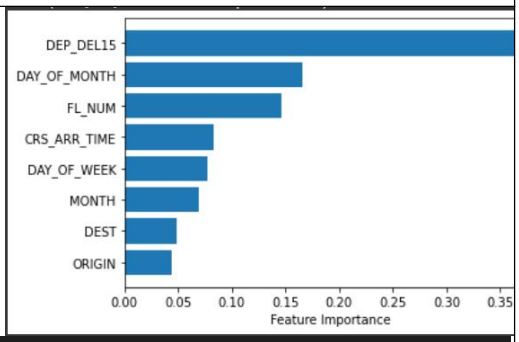
1.	Model Summary	Random Forest Classifier	Random Forest Classifier
			`forest_reg = RandomForestClassifier(n_estimators = 10, criterion = 'entropy',random_state`
			`forest_reg.fit(X_train,Y_train)`
			`model = forest_reg.fit(X_train , Y_train.ravel())`
			`pred = model.predict(X_test)`
			`print('Mean Absolute Error:', metrics.mean_absolute_error(Y_test, pred))`
			`print('Mean Squared Error:', metrics.mean_squared_error(Y_test, pred))`
			`print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(Y_test, pred)))`
			`print("Accuracy:",metrics.accuracy_score(Y_test, pred))`
			`#print("Report for test size = ", test_sz[i], "is shown below:")`
			`print(classification_report(Y_test, pred))`
			`matrix_confusion = confusion_matrix(y_true=Y_test, y_pred=pred)`
			`print(matrix_confusion)`
			`sns.heatmap(matrix_confusion, square=True, annot=True, cmap='Blues', fmt='d', cbar=False)`

2.	Accuracy	Training Accuracy - 89
		Classification Report
			`Mean Absolute Error: 0.10007122507122507`
			`Mean Squared Error: 0.10007122507122507`
			`Root Mean Squared Error: 0.3163403626969298`
			`Accuracy: 0.8999287749287749`
			`precision recall f1-score support`
			`0.0 0.93 0.96 0.94 2421`
			`1.0 0.68 0.52 0.59 387`
			`accuracy 0.90 2808`
			`macro avg 0.80 0.74 0.77 2808`
			`weighted avg 0.89 0.90 0.89 2808`

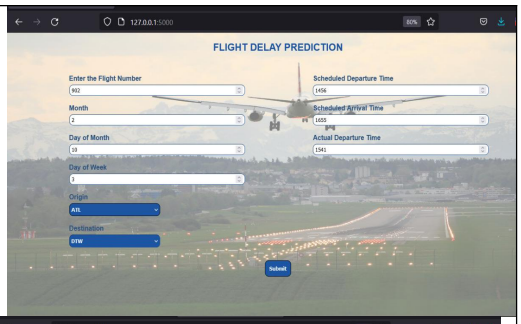
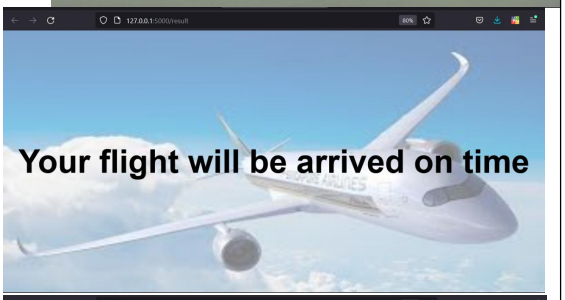
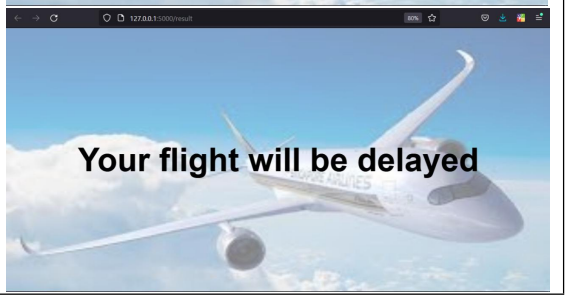
3.	Metrics	Confusion Matrix	

4.	Label Encoding	Encoding Results	
5.	Feature Selection	Random Forest Regressor	

	FL_NUM	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	ORIGIN	DEST	CRS_ARR_TIME	DEP_DEL15	ARR_D
0	1399	1	1	5	0	4	21	0.0	
1	1476	1	1	5	1	3	14	0.0	
2	1597	1	1	5	0	4	12	0.0	
3	1768	1	1	5	4	3	13	0.0	
4	1823	1	1	5	4	1	6	0.0	




```
from sklearn.ensemble import RandomForestRegressor
from sklearn.inspection import permutation_importance
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split

dataset.replace([np.inf, -np.inf], np.nan, inplace=True)
dataset.fillna(999, inplace=True)

X = dataset.iloc[:, 0:8].values
Y = dataset.iloc[:, 8:9].values

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=42)

rf = RandomForestRegressor(n_estimators=150)
rf.fit(X_train, Y_train)

sort = rf.feature_importances_.argsort()
print(rf.feature_importances_)
plt.barh(dataset.columns[sort], rf.feature_importances_[sort])
plt.xlabel("Feature Importance")
```

6.	UI	Dashboard Design	  

10 ADVANTAGES & DISADVANTAGES

ADVANTAGES:

This application is helpful for both the flight management the business professionals and frequent air travelers . It helps them to identify whether their flight will be on time or delayed . So they can reschedule their important meetings or other work to some other time.

DISADVANTAGES:

Even though it predicts the delay of flight . It cannot tell exactly . Sometimes due to some unexpected natural calamities ,flight may be delayed . So this application is not 100 percent accurate .

11 CONCLUSION:

Due to increase in flight travel nowadays ,the air traffic is significantly increasing. Hence so many measures have been taken by the air travel companies to reduce this . Because of advancement in technology now they are implementing these application using machine learning to get expected outcomes.

12 FUTURE SCOPE:

In future this application can be further improved by implementing block-chain and some advanced techniques like neural networks . Further it can be improved by using live data instead of previous record of flight.

13 APPENDIX:

App.py

```
from flask import Flask, render_template, request
import pandas as pd
import joblib
import numpy as np

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('Flightdelay.html')

@app.route('/result', methods=['POST'])
def predict():
    fl_num = int(request.form.get('fno'))
    month = int(request.form.get('month'))
    dayofmonth = int(request.form.get('daym'))
    dayofweek = int(request.form.get('dayw'))
    origin = str(request.form.get("org"))
    dest = str(request.form.get("dest"))
    sdeptime = request.form.get('sdt')
    sarrtime = int(request.form.get('sat'))
    adeptime = request.form.get('adt')

    inputs = list()
    inputs.append(fl_num)
    inputs.append(month)
    inputs.append(dayofmonth)
    inputs.append(dayofweek)
    inputs.append(sdeptime)
    inputs.append(sarrtime)
    inputs.append(adeptime)
```

```
    if(origin == "ATL"):
        a = [1, 0, 0, 0, 0]
        inputs.extend(a)
    elif(origin == "DTW"):
        a = [0, 1, 0, 0, 0]
        inputs.extend(a)
    elif(origin == "JFK"):
        a = [0, 0, 1, 0, 0]
        inputs.extend(a)
    elif(origin == "MSP"):
        a = [0, 0, 0, 1, 0]
        inputs.extend(a)
    elif(origin == "SEA"):
        a = [0, 0, 0, 0, 1]
        inputs.extend(a)

    if(dest == "ATL"):
        b = [1, 0, 0, 0, 0]
        inputs.extend(b)
    elif(dest == "DTW"):
        b = [0, 1, 0, 0, 0]
        inputs.extend(b)
    elif(dest == "JFK"):
        b = [0, 0, 1, 0, 0]
        inputs.extend(b)
    elif(dest == "MSP"):
        b = [0, 0, 0, 1, 0]
        inputs.extend(b)
    elif(dest == "SEA"):
        b = [0, 0, 0, 0, 1]
        inputs.extend(b)

    # inputs.append(origin)
    # inputs.append(dest)

    print(inputs)
    prediction = preprocessAndPredict(inputs)
    # Pass prediction to prediction template
    return render_template('/result.html', prediction=prediction)

def preprocessAndPredict(inputs):
    test_data = np.array(inputs).reshape((1, 17))
    # test_data = inputs

    model_file = open('model_forest_reg.pkl', 'rb')

    trained_model = joblib.load(model_file)
```

```
   # df = pd.DataFrame(data=test_data[0:, 0:], columns=['FL_NUM', 'MONTH',
'DAY_OF_MONTH','DAY_OF_WEEK', 'origin', 'dest', 'CRS_DEP_TIME', 'CRS_ARR_TIME', 'DEP_TIME'])

   df = pd.DataFrame(data=test_data[0:, 0:], columns=['FL_NUM', 'MONTH', 'DAY_OF_MONTH',
'DAY_OF_WEEK', 'CRS_DEP_TIME', 'CRS_ARR_TIME', 'DEP_TIME',
                  'ORIGIN_ATL', 'ORIGIN_DTW', 'ORIGIN_JFK', 'ORIGIN_MSP', 'ORIGIN_SEA', 'DEST_ATL',
'DEST_DTW', 'DEST_JFK', 'DEST_MSP', 'DEST_SEA'])

   data = df.values

   result = trained_model.predict(data)

   print(result)
   return result

if __name__ == '__main__':
   app.run(debug=True)
```

UI*:*

```
<!DOCTYPE html>

<html lang="en">

<head>

        <meta charset="UTF-8">

        <meta http-equiv="X-UA-Compatible">

        <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <link rel="stylesheet" href="{{ url_for('static',filename='styles/styles.css') }}">

        <script src="{{url_for('static', filename='styles/delaypredict.js')}}"></script>

        <title>Flight Delay Prediction</title>

</head>

<body id="flight-form">

<h2 id="main-head" class="centered-head">FLIGHT DELAY PREDICTION</h2>

<img src="{{url_for('static', filename='styles/images/Flight1.jpg')}}" id="bgimg">

<form name="flightForm" action="/result" method="POST" target="_blank">

        <div id="form-content">

                <div id="block1">
```

```
                    <div class="detail-container">
                        <label for="fno" class="label-item">Enter the Flight Number</label>
                        <br>
                        <input type="number" id="fno" name="fno" class="text-input">
                    </div>
                    <div class="detail-container">
                        <label for="month" class="label-item">Month</label>
                        <br>
                        <input type="number" id="month" name="month" class="text-
input" onblur="checkValid('month');" placeholder="Enter the Month Number">
                        <div class="alert-text" id="month-valid">Enter a valid month
between 1 to 12.</div>
                    </div>
                    <div class="detail-container">
                        <label for="daym" class="label-item">Day of Month</label>
                        <br>
                        <input type="number" id="daym" name="daym" class="text-input"
onblur="checkValid('daym');">
                        <div class="alert-text" id="daym-valid">Enter a valid day of
month.</div>
                    </div>
                    <div class="detail-container">
                        <label for="dayw" class="label-item">Day of Week</label>
                        <br>
                        <input type="number" id="dayw" name="dayw" class="text-input"
onblur="checkValid('dayw');">
                        <div class="alert-text" id="dayw-valid">Enter a valid day between 1
to 7.</div>
                    </div>
```

```
                    <div class="detail-container">
                        <label for="org" class="label-item">Origin</label>
                        <br>
                        <select id="org" name="org" class="select-input">
                            <option value="ATL" class="option-item">ATL</option>
                            <option value="SEA" class="option-item">SEA</option>
                            <option value="DTW" class="option-item">DTW</option>
                            <option value="MSP" class="option-item">MSP</option>
                            <option value="JFK" class="option-item">JFK</option>
                        </select>
                    </div>
                    <div class="detail-container">
                        <label for="dest" class="label-item">Destination</label>
                        <br>
                        <select id="dest" name="dest" class="select-input"
onblur="checkValid('dest');">
                            <option value="ATL" class="option-item">ATL</option>
                            <option value="SEA" class="option-item">SEA</option>
                            <option value="DTW" class="option-item">DTW</option>
                            <option value="MSP" class="option-item">MSP</option>
                            <option value="JFK" class="option-item">JFK</option>
                        </select>
                        <div class="alert-text" id="dest-valid">Enter different Origin and
Destination.</div>
                    </div>
                </div>
                <div id="block2">
```

```
                    <div class="detail-container">
                        <label for="sdt" class="label-item">Scheduled Departure Time</label>
                        <br>
                        <input type="number" id="sdt" name="sdt" class="text-input" onblur="checkValid('sdt');" placeholder="Enter in the format HHMM">
                        <div class="alert-text" id="sdt-valid">Enter a valid time between 500 to 2359.</div>
                    </div>
                    <div class="detail-container">
                        <label for="sat" class="label-item">Scheduled Arrival Time</label>
                        <br>
                        <input type="number" id="sat" name="sat" class="text-input" onblur="checkValid('sat');" placeholder="Enter in the format HHMM">
                        <div class="alert-text" id="sat-valid">Enter a valid time between 500 to 2359.</div>
                    </div>
                    <div class="detail-container">
                        <label for="adt" class="label-item">Actual Departure Time</label>
                        <br>
                        <input type="number" id="adt" name="adt" class="text-input" onblur="checkValid('adt');" placeholder="Enter in the format HHMM">
                        <div class="alert-text" id="adt-valid">Enter a valid time between 500 to 2359.</div>
                    </div>
                </div>
            </div>
            <div id="submit-button">
                <input type="submit" value="Submit" id="submit" class="button" onclick="validateForm()">
```

```
        </div>
</form>
</body>
</html>
```

GITHUB AND PROJECT DEMO LINK

Github link:

https://github.com/IBM-EPBL/IBM-Project-1927-1658420908/tree/main/Final%20Deliverables