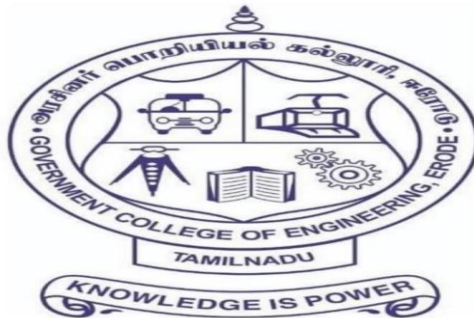


GOVERNMENT COLLEGE OF ENGINEERING
(Formerly IRTT)
ERODE-638 316



Certified that this project titled **“DEVELOPING A FLIGHT DELAY PREDICTION USING MACHINE LEARNING”** is the bonafide work of **"PARKKAVI T(731119104036), PRAVENA S(731119104039), SALLY JEMIMAH J(731119104041), VAISHNAVI A(731119104050)"** who carried out the project work under my supervision.

SIGNATURE OF HOD

Dr.A.SARADHA,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
DEPARTMENT OF CSE,
GOVERNMENT COLLEGE OF
ENGINEERING, ERODE – 638316

SIGNATURE OF SPOC

Dr.G.GOWRISON, M.E.,Ph.D.,
ASSISTANT PROFESSOR(SR)
DEPARTMENT OF ECE,
GOVERNMENT COLLEGE OF
ENGINEERING, ERODE - 638316

SIGNATURE OF FACULTY MENTOR

Dr.A.KAVIDHA,M.E.Ph.D.,
ASSOCIATE PROFESSOR
DEPARTMENT OF CSE,
GOVERNMENT COLLEGE OF
ENGINEERING, ERODE-- 638316 .

**SIGNATURE OF FACULTY
EVALUATOR**

Mrs.N.VASUKI,M.E.,
ASSISTANT PROFESOR
DEPARTMENT OF CSE,
GOVERNMENT COLLEGE OF
ENGINEERING, ERODE-638316.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE NO
1.	INTRODUCTION	
	1.1 PROJECT OVERVIEW	1
	1.2 PURPOSE	1
2.	LITERATURE SURVEY	
	2.1 EXISTING PROBLEM	2
	2.2 REFERENCES	2
	2.3 PROBLEM STATEMENT DEFINITION	2
3.	IDEATION AND PROPOSED SOLUTION	
	3.1 EMPATHY MAP CANVAS	3
	3.2 IDEATION AND BRAINSTOMING	4
	3.3 PROPOSED SOLUTION	5
	3.4 PROBLEM SOLUTION FIT	6
4.	REQUIREMENT ANALYSIS	
	4.1 FUNCTIONAL REQUIREMENT	7
	4.2 NON FUNCTIONAL REQUIREMENT	7
5.	PROJECT DESIGN	
	5.1 DATA FLOW DIAGRAM	9
	5.2 SOLUTION AND TECHNICAL ARCHITECTURE	10
	5.3 USER STORIES	13
6.	PROJECT PLANNING AND SCHEDULING	
	6.1 SPRINT PLANNING AND ESTIMATION	14
	6.2 SPRINT DELIVERY SCHEDULE	15
	6.3 REPORTS FROM JIRA	17

7.	CODING AND SOLUTIONING	
	7.1 FEATURE 1	18
	7.2 FEATURE 2	20
	7.3 DATABASE SCHEMA	22
8.	TESTING	
	8.1 TEST CASES	23
	8.2 USER ACCEPTANCE TESTING	24
9.	RESULTS	
	9.1 PERFORMANCE METRICS	26
10.	ADVANTAGES AND DISADVANTAGES	29
11.	CONCLUSION	30
12.	FUTURE SCOPE	30
13.	APPENDIX	
	SOURCE CODE	31
	GITHUB AND PROJECT DEMO LINK	61

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.

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.

1.2 PURPOSE

Flight delay is inevitable and it plays an important role in both profits and loss of the airlines. An accurate estimation of flight delay is critical for airlines because the results can be applied to increase customer satisfaction and incomes of airline agencies.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

While airlines want to get passengers to their destinations on time, there are many things that can – and sometimes do – make it difficult for flights to arrive on time. Some problems, like bad weather, air traffic delays, and mechanical issues, are hard to predict and often beyond the airlines' control.

2.2 REFERENCES

Yuemin Tang.2021.Airline Flight Delay Prediction Using Machine Learning Models.In 2021 5th International Conference on E-Business and Internet (ICEBI), October 15-17,2021,Singapore,Singapore.ACM,New York, NY, USA, 7 Pages.
<https://doi.org/10.1145/3497701.3497725>

Bhuvan Bhatia used python based Logistic Regression along with Support Vector Machine to predict flight delays and compared the results with other models such as RandomForest Classifier and derive the best classifier to solve the problem. The dataset focused on LaGuardia International Airport. The result shows that the Random Forest method yields the best performance compared to the SVM model.

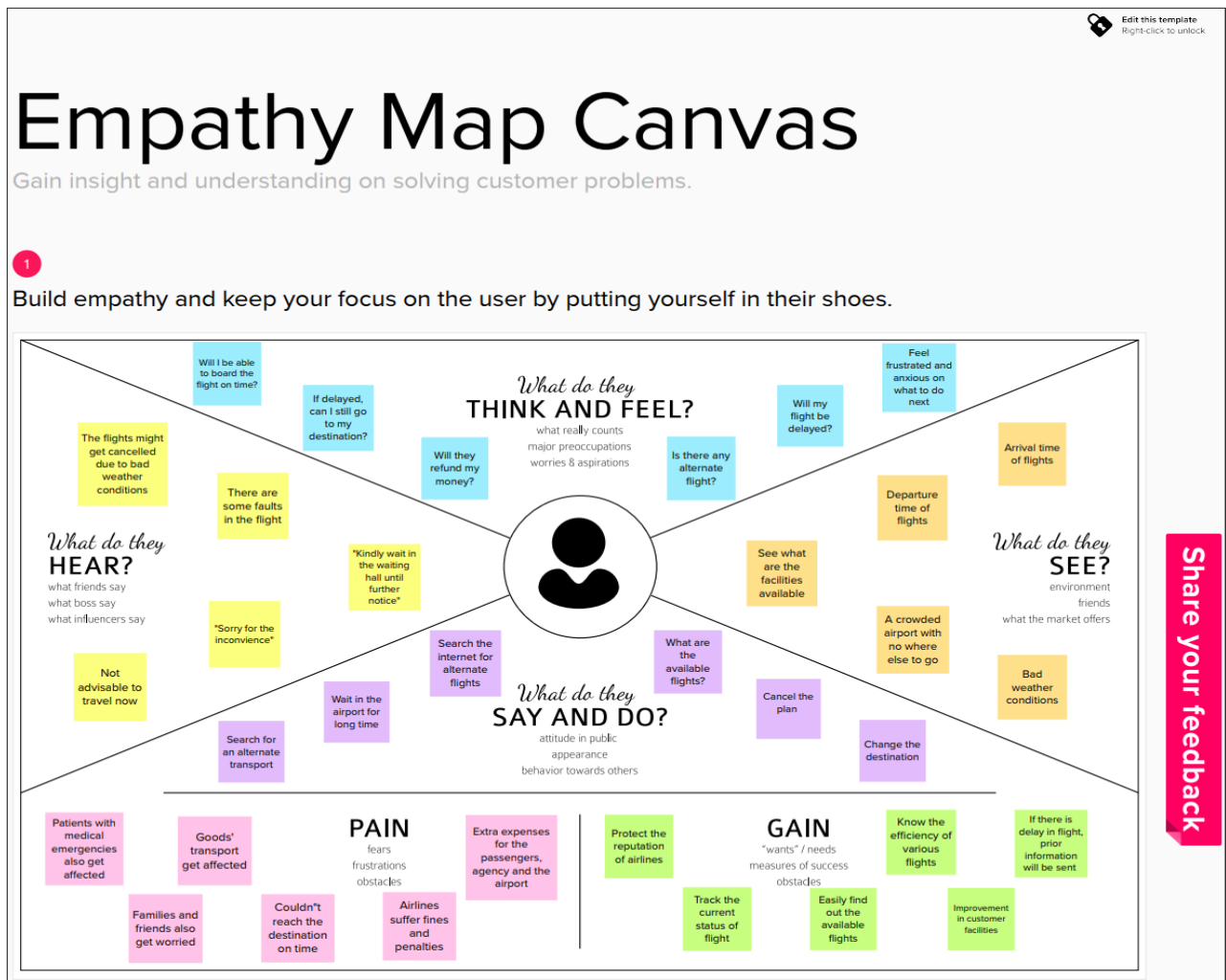
<https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/itr2.12183>
<https://medium.com/analytics-vidhya/using-machine-learning-to-predict-flight-delays-e8a50b0bb64c>

2.3 PROBLEM STATEMENT DEFINITION

Flight Delay Prediction uses certain features of the flight like the airlines who operate them, distance they have to cover ,origin and target airport ,arrival and departure times and so on .It can help the passengers to know what delays they should be ready to face depending on where they fly from the airlines they choose to fly. It is of immense help to the passengers, airlines and the aviation

3. IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION AND PROPOSED SOLUTION

Microsoft Edge is now your default PDF reader. [Learn more](#) X

Brainstorm & idea prioritization

Use this template in your next brainstorming sessions so your team can unleash their imagination and start shaping concepts when you're not sitting in the same room.

- 1. Welcome to your
- 2. Ideas generated

Before you collaborate

Write down all your ideas on sticky notes and stick them on the board. You can use the board to organize your ideas.

[Go to slide 2](#)

Define your problem statement

What problem are you trying to solve? Frame your problem as a challenge. Write down the problem statement. You will use this to guide your brainstorming.

[Go to slide 3](#)

Brainstorm

Write down any ideas that come to mind that address your problem statement.

[Go to slide 4](#)

Group ideas

Take your ideas and group them into clusters or related ideas as you go. In the end, you will have a collection of ideas that you can use to develop your solution.

[Go to slide 5](#)

Profile

You need to think about the same problem statement that you've been working on. Think about the problem statement that you've been working on and what you have learned.

[Go to slide 6](#)

After you collaborate

You can export the board as an image or PDF to share with members of your company who might find it helpful.

[Go to slide 7](#)

Brainstorm & idea prioritization

Use this template in your next brainstorming sessions so your team can unleash their imagination and start shaping concepts when you're not sitting in the same room.

- 1. Welcome to your
- 2. Ideas generated

Before you collaborate

Write down all your ideas on sticky notes and stick them on the board. You can use the board to organize your ideas.

[Go to slide 2](#)

Define your problem statement

What problem are you trying to solve? Frame your problem as a challenge. Write down the problem statement. You will use this to guide your brainstorming.

[Go to slide 3](#)

Brainstorm

Write down any ideas that come to mind that address your problem statement.

[Go to slide 4](#)

Group ideas

Take your ideas and group them into clusters or related ideas as you go. In the end, you will have a collection of ideas that you can use to develop your solution.

[Go to slide 5](#)

Profile

You need to think about the same problem statement that you've been working on. Think about the problem statement that you've been working on and what you have learned.

[Go to slide 6](#)

After you collaborate

You can export the board as an image or PDF to share with members of your company who might find it helpful.

[Go to slide 7](#)

3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The main objective of the model is to predict flight delays accurately in order to optimize flight operations, to save passengers and airlines from all the hardships caused due to flight delays or in worst case cancellations
2.	Idea / Solution description	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.
3.	Novelty / Uniqueness	We compare decision tree classifier with logistic regression and a simple neural network for various figures of merit.
4.	Social Impact / Customer Satisfaction	Time management will be the social impact such that when informed earlier the passengers can plan accordingly.
5.	Business Model (Revenue Model)	* Low-cost airline business model. * B2C business Model.
6.	Scalability of the Solution	The delayed time of any type of flight can be known with maximum accuracy

3.4 PROBLEM SOLUTION FIT

1. CUSTOMER SEGMENT Business peoples and regular flight users	6. CUSTOMER CONSTRAINTS Lack of transparency, no user-friendly models to work with	5. AVAILABLE SOLUTIONS Weather forecasting, creation of larger runways, effective air traffic control
2. JOBS-TO-BE-DONE/ PROBLEMS Predicting the flight delay due to the various reasons that may cause it, Intimate the flight delay to the passengers, Provide alternate flights, if the delay is prolonged	9. PROBLEM ROOT / CAUSE Adverse weather conditions, air traffic, bird strikes, less runways, waiting for connecting passengers and bags, flight malfunction	7. BEHAVIOUR Choose the right airlines, Choose different modes of transport, Wait patiently in the waiting hall until further notification, Search online for alternate flights, Dissatisfied and frustrated
3. TRIGGERS Seeing other airlines that give accurate departure and arrival time even with delay 4. EMOTIONS: before /after Frustration -> Satisfaction	10. YOUR SOLUTION By using machine learning algorithms we can try to predict if the flight will be delayed in many ways. If given the right set of input parameters (Flight no, departure and arrival time, origin and destination airport, scheduled arrival and departure time, etc.), the ML algorithms can predict the delay with high accuracy	8. CHANNELS OF BEHAVIOUR 8.1 ONLINE Check for reimbursements, Search for the right airlines, book alternate flights online, agree to a new connection, call the airline 8.2 OFFLINE Don't plan activities on the day of arrival, schedule flights for the middle of the week, fly non-stop routes, avoid travelling during holidays

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User registration & login	Registration & login of passengers via Google with email id and password
FR-2	Detailed arrival and departure time of flights	With the flight no and name, the passenger can see the details (time, boarding station, etc) of his/her in the dashboard.
FR-3	Intimate the accurate flight timings to passengers	With the help of various machine learning algorithms, when given the right input features (actual arrival time & departure time, scheduled time, etc) we can predict the delay in time of the flight which will also be shown in the dashboard and updated time-to-time.
FR-4	Airline helpdesk provide alternatives	The contact details of different airlines will be provided, The passenger will also be able to look for any alternative flight in case the flights get cancelled.
FR-5	Passenger feedback	The feedback will be got from the users or how the application was to use, with their feedback and suggestions, we can improve the application further.

4.2 NON FUNCTIONAL REQUIREMENT

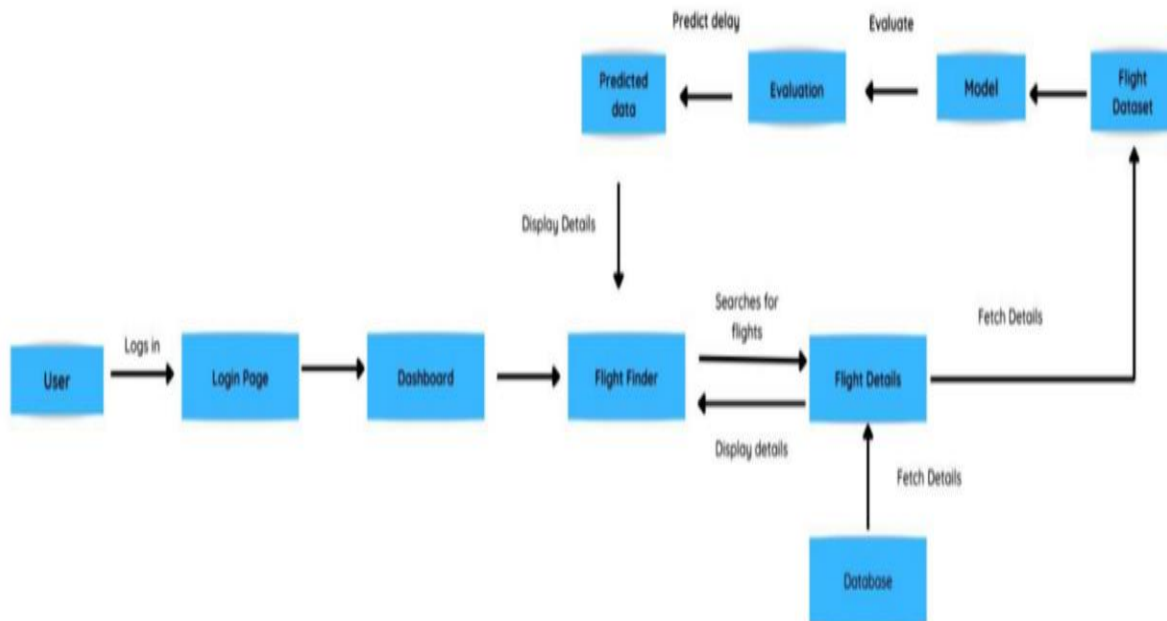
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The application will have an easy-to-use GUI. Users will find it simple to comprehend and utilize all the capabilities of the application.
NFR-2	Security	The technique known as database replication will be utilised for the application security to ensure the safety of all crucial data
NFR-3	Reliability	The application will be consistent in all scenarios and work without fail in any environment
NFR-4	Performance	The applications response time is direct & faster which is determined by the efficiency of the implemented machine algorithm.

NFR-5	Availability	The application will be accessible to users 24 hours a day, 7 days a week without interruption. They can access it from any part of the world with proper internet.
NFR-6	Scalability	The application will be able to handle a rise in the no. of users & generate higher versions.

5. PROJECT DESIGN

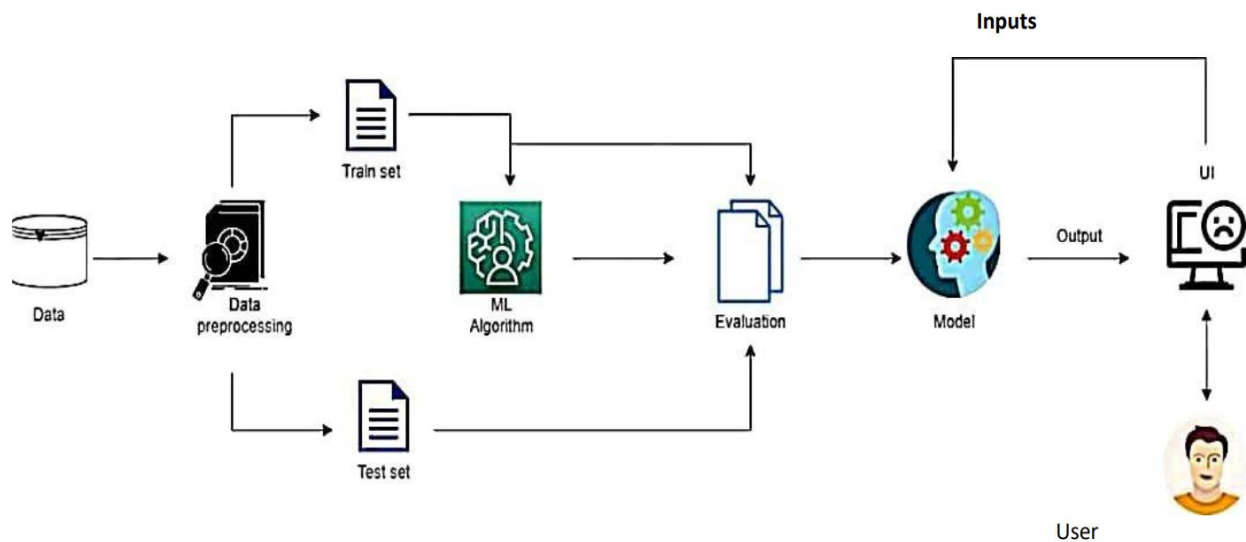
5.1 DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 SOLUTION AND TECHNICAL ARCHITECTURE

SOLUTION ARCHITECTURE



TECHNICAL ARCHITECTURE

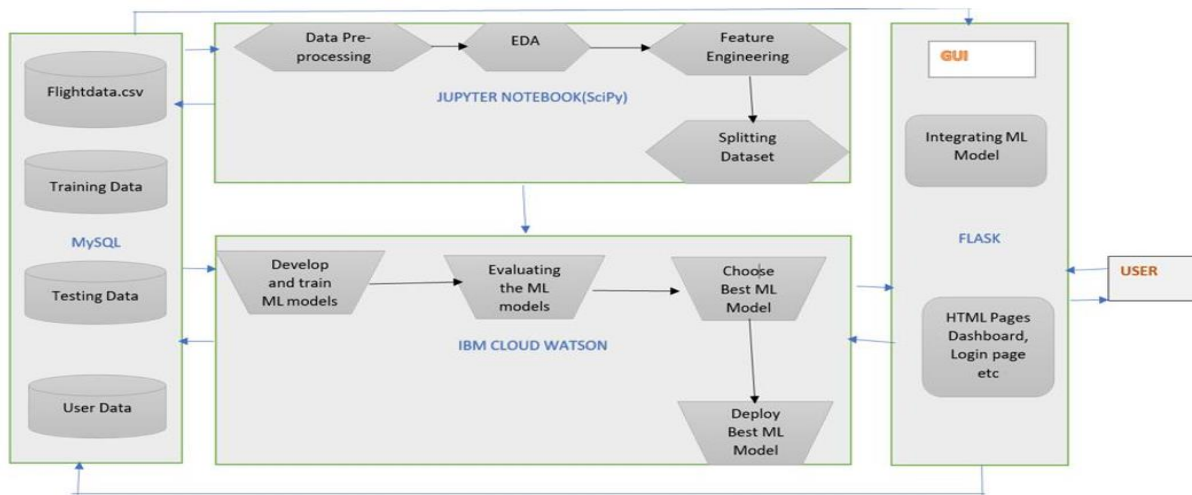


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application	HTML, CSS, Flask
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Data processing	To clean data	Pandas, NumPy, Matplotlib etc.,
5.	Database	Store data	MySQL
6.	File Storage	Storing files	IBM Block Storage
7.	External API-1	External API used in the system	IBM Weather API
8.	External API-2	External API used in the system	Email API
9.	Machine Learning Model	Purpose of Machine Learning Model	Evaluation and prediction model
	Infrastructure (Server / Cloud)	Application Deployment	IBM Cloud

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source software is that by which the source code or the base code is usually available for modification or enhancement	Python - flask
2.	Security Implementations	The security measures can be grouped into two types; standardized screening techniques, which all passengers must undergo and elevate-risk	Encryptions, SHA2
3.	Scalable Architecture	Screening for which only a subset of passengers are selected	Python
4.	Availability	Does not affect the performance even though used by many users	IBM cloud
5.	Performance	High delay prediction accuracy	Python, Flask

5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile or Web user)	Registration and Login	USN-1	As a new user, I can register for the application by entering my email and my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	I will receive a confirmation message through email after registration	I can receive confirmation email & click confirm	Medium	Sprint-1
		USN-3	As a user, I can log into the application by entering the registered email-id and password	I can register & access the dashboard with Gmail Login	High	Sprint-1
Customer (Mobile or Web user)	Arrival and Departure time of flights	USN-4	As a user, I can search for the details of a specific flight with flight number or name	I can find all the details of a flight	Medium	Sprint-2
		USN-5	As a user, I can find the accurate arrival and departure time of flights	I can find the actual timings of the flight	High	Sprint-3
Customer (Mobile or Web user)	Real time flight delay	USN-6	As a user, I can find exactly how long the flight will be delayed	I can get the accurate delayed time	High	Sprint-3
		USN-7	As a user, I can get real time timings that are updated every few seconds.	I can check the updated time	High	Sprint-3
Customer Care Executive	Helpdesk	USN-8	I can provide other alternative flights to the passenger's destination	I can check for alternative flights	High	Sprint-2
		USN-9	As a customer care executive, I can provide the contact details of the airlines to help the passenger to get in touch with them in case of any query	I can give the airlines' phone number	Medium	Sprint-4
Customer Care Executive	Feedback	USN-10	I can collect all the feedback and suggestions that are given by the passengers, after using this application	I am able to record the feedbacks	Medium	Sprint-4
Administrator	Authentication	USN-11	As an admin, I can authenticate the registration and login credentials of the passengers.	I can validate the passengers' login	High	Sprint-1
		USN-12	As an admin, I ensure the security of the passengers' details	I maintain the security of user details	High	Sprint-4

6.PROJECT PLANNING AND SCHEDULING

6.1 SPRINT PLANNING AND ESTIMATION

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration and Login	USN-1	As a new user, I can register for the application by entering my email and my password.	10	High	Pravena S
Sprint-2		USN-2	As a user, I will receive confirmation email once I have registered for the application	10	Medium	Parkkavi T Pravena S
Sprint-1		USN-3	As a user, I can login into the application by entering the registered email-id and password	10	High	Vaishnavi A Sally Jemimah J
Sprint-2	Admin Panel	USN-4	As an admin, I can authenticate the registration and login credentials of the passengers	10	High	Pravena S Vaishnavi A Parkkavi T Sally Jemimah J
Sprint-3	Arrival and Departure time of flights	USN-5	As a user, I can find all the details of a specific flight with its number or name	10	High	Pravena S Vaishnavi A Parkkavi T Sally Jemimah J
Sprint-3		USN-6	As a user, I can find exactly how long the flight will be delayed	10	High	Pravena S Vaishnavi A Parkkavi T Sally Jemimah J
Sprint-4	Helpdesk	USN-7	As a customer care executive, I can provide the contact details of the airlines	05	Medium	Vaishnavi A Pravena S
Sprint-4		USN-8	As a passenger, I can find alternative flights to the destination that are available	05	High	Parkkavi T Sally Jemimah J
Sprint-4	Feedback	USN-9	As a user, I can provide my suggestions and feedback for the improvement of the application	10	Medium	Sally Jemimah J Parkkavi T Vaishnavi A

6.2 SPRINT DELIVERY SCHEDULE

Project Tracker, Velocity & Burndown Chart:

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	05 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

Velocity:

We have a 24-day sprint duration, and the velocity of the team is 20 (points per sprint). Thus the team's average velocity (AV) per iteration unit (story points per day) is as follows

$$AV = \text{Sprint duration} / \text{Velocity}$$

$$= 24/20$$

$$= 1.2$$

Average velocity = $6/20 = 0.3$ (Sprint 1)

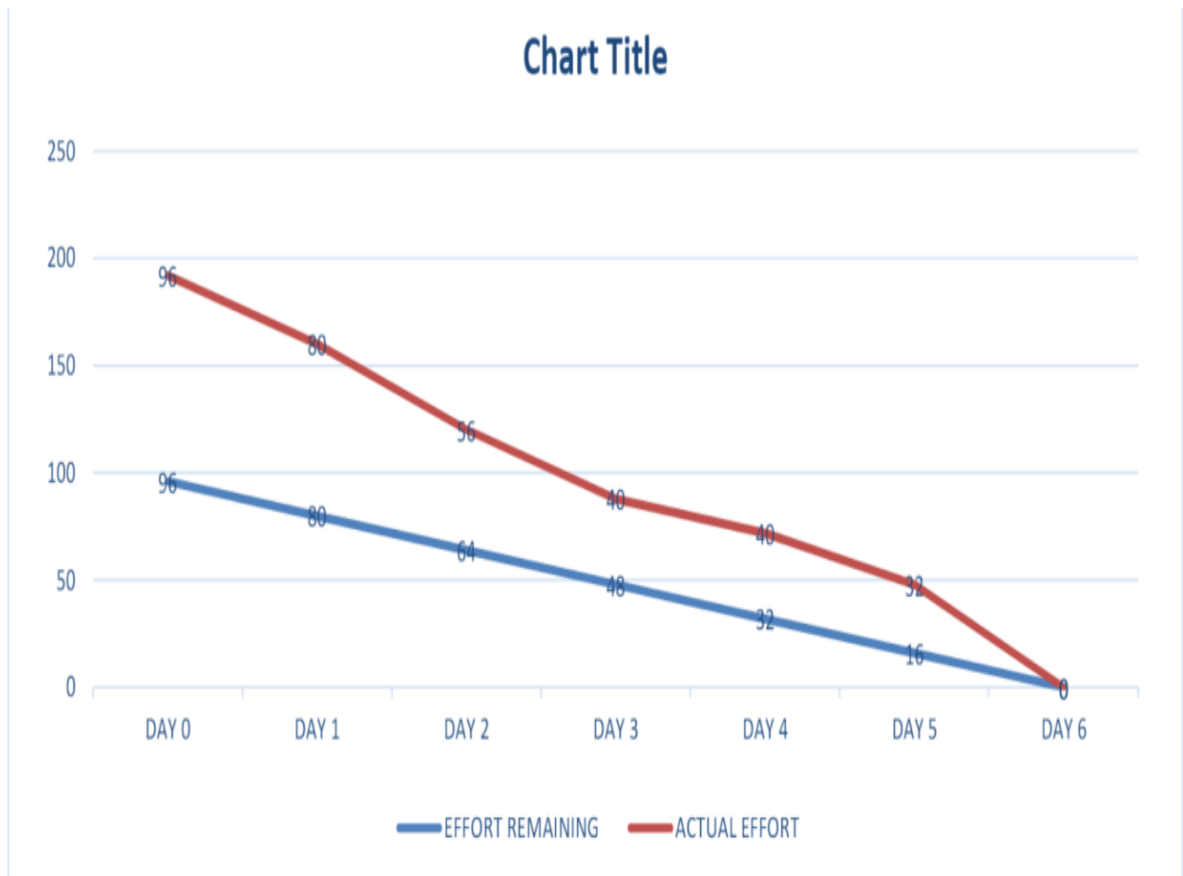
Average velocity = $6/20 = 0.3$ (Sprint 2)

Average velocity = $6/20 = 0.3$ (Sprint 3)

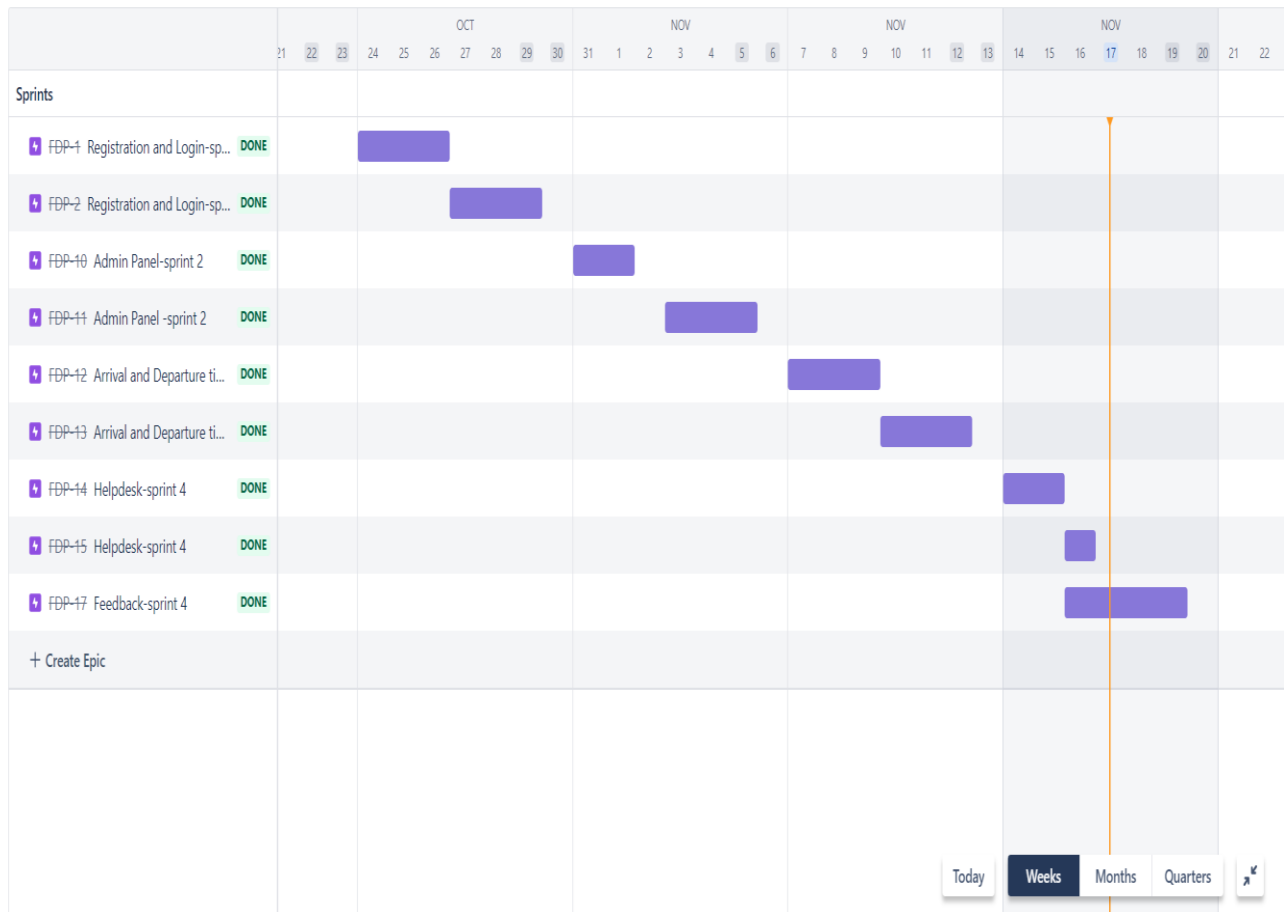
Average velocity = $6/20 = 0.3$ (Sprint 4)

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum.



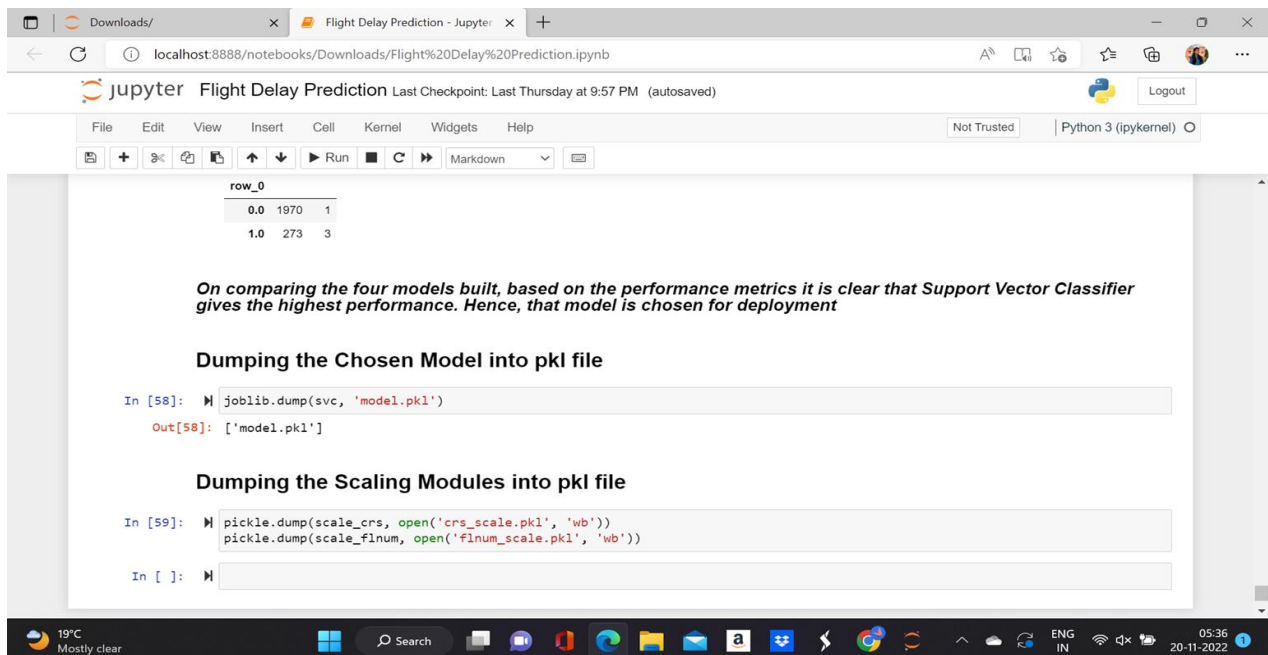
6.3 REPORTS FROM JIRA



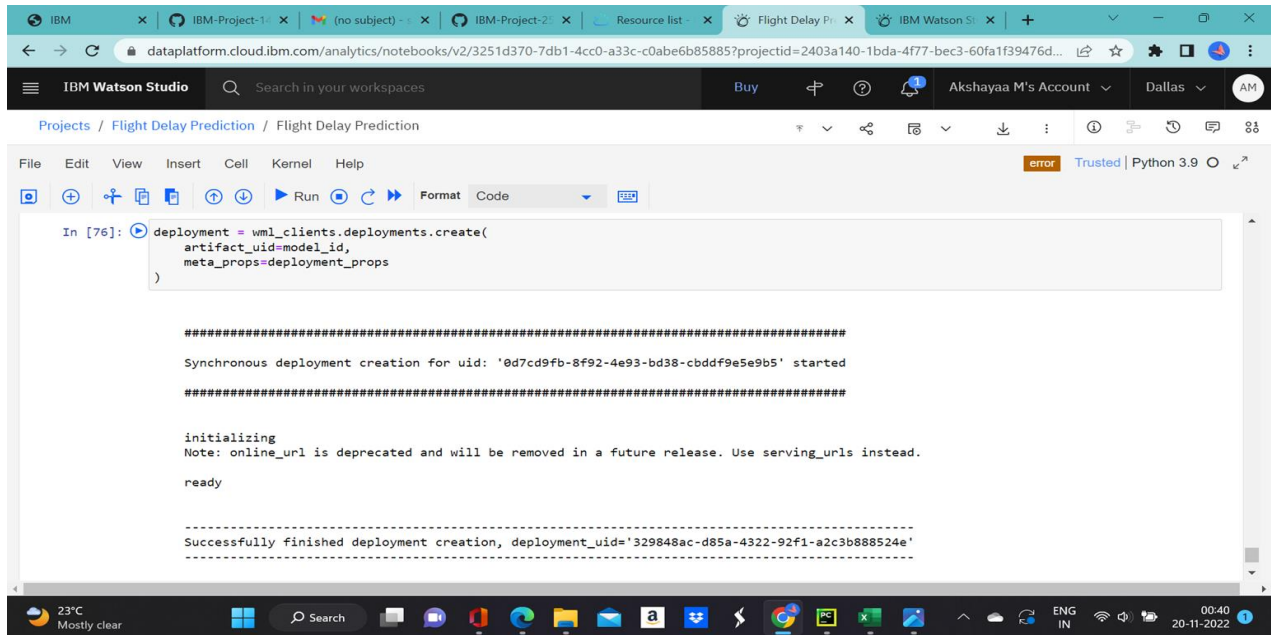
7. CODING AND SOLUTIONING

7.1 FEATURE 1

Step 1: The flightdata.csv was preprocessed and the machine learning models were built using jupyter notebook



Step 2: The ML model was then deployed in IBM Watson and the API key and endpoint were noted down



The screenshot shows the IBM Watson Studio interface. The browser address bar displays the URL: `dataplatfom.cloud.ibm.com/analytics/notebooks/v2/3251d370-7db1-4cc0-a33c-c0abe6b85885?projectId=2403a140-1bda-4f77-bec3-60fa1f39476d...`. The notebook cell contains the following Python code:

```
In [76]: deployment = wml_clients.deployments.create(
          artifact_uid=model_id,
          meta_props=deployment_props
        )
```

The output of the cell shows the deployment process:

```
#####

Synchronous deployment creation for uid: '0d7cd9fb-8f92-4e93-bd38-cbdf9e5e9b5' started

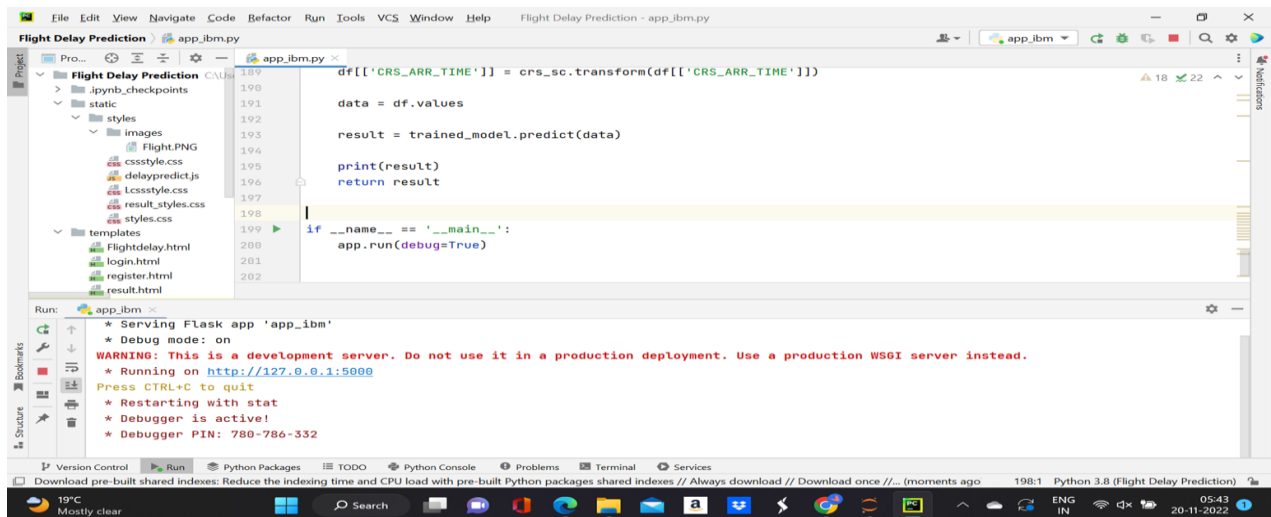
#####

initializing
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.

ready

-----
Successfully finished deployment creation, deployment_uid='329848ac-d85a-4322-92f1-a2c3b888524e'
-----
```

Step 3: The python flask code was written and run to connect the html pages with the deployed ML model in IBM Watson



The screenshot shows a Python Flask application running in a terminal window. The file explorer on the left shows the project structure for 'Flight Delay Prediction', including files like `Flight.PNG`, `cssstyle.css`, `delaypredict.js`, `lcsstyle.css`, `result_styles.css`, `styles.css`, `templates`, `Flightdelay.html`, `login.html`, `register.html`, and `result.html`. The main code file is `app_ibm.py`, which contains the following Python code:

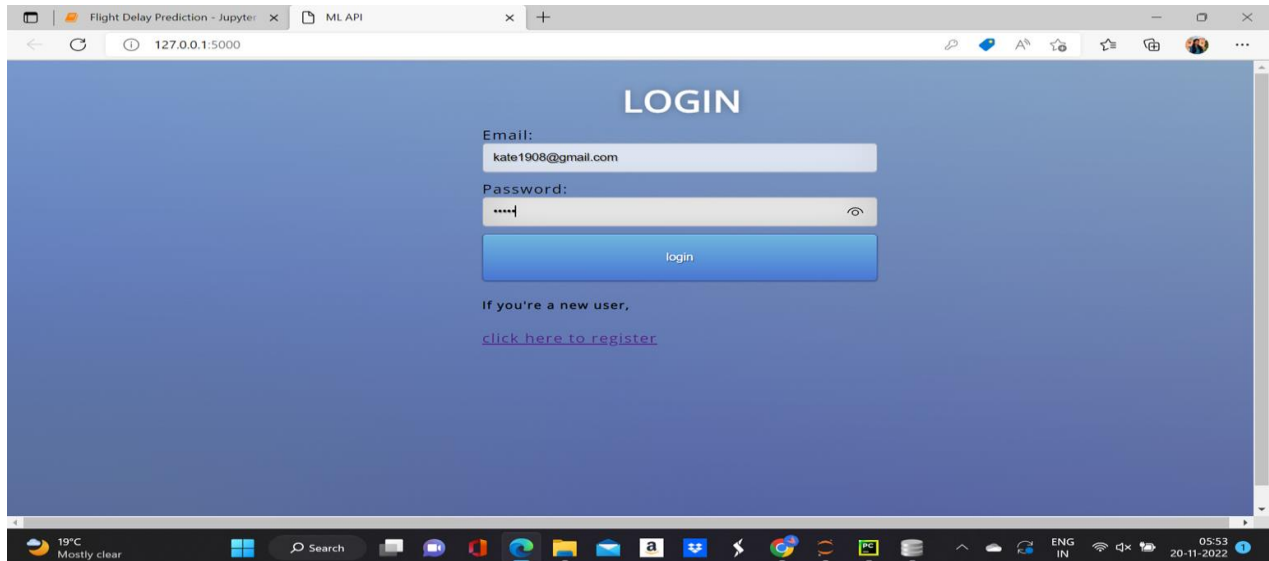
```
189 df[['CRS_ARR_TIME']] = crs_sc.transform(df[['CRS_ARR_TIME']])
190
191 data = df.values
192
193 result = trained_model.predict(data)
194
195 print(result)
196
197 return result
198
199 if __name__ == '__main__':
200     app.run(debug=True)
201
202
```

The terminal output shows the Flask application running on `http://127.0.0.1:5000`. The output includes the following messages:

```
* Serving Flask app 'app_ibm'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 780-786-332
```

7.2 FEATURE 2

Step 1: If the user has already registered, then he/she can log into the application by entering the correct email-id and password

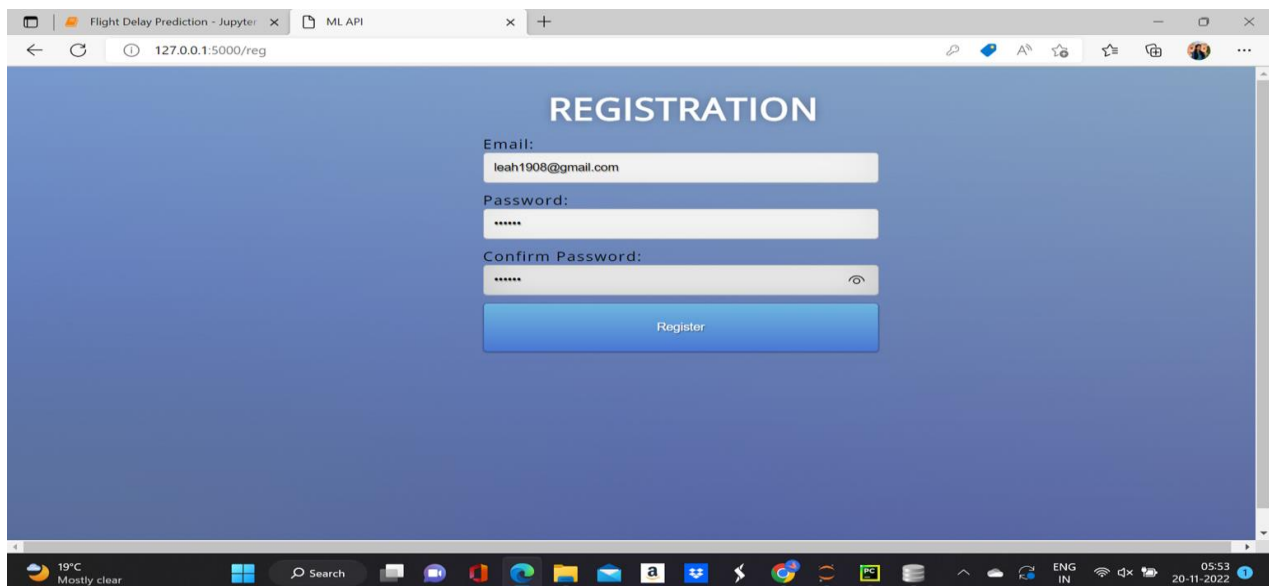


The screenshot shows a web browser window with the title "Flight Delay Prediction - Jupyter" and "ML API". The address bar shows "127.0.0.1:5000". The main content area is titled "LOGIN" and contains a form with the following fields:

- Email:
- Password:
- A blue "login" button.

Below the form, there is a link: "If you're a new user, [click here to register](#)". The browser's taskbar at the bottom shows the date and time as 05:53 on 20-11-2022.

Step 2: If the user is new, then he/she should have to register first. After registration, they can be logged into the application

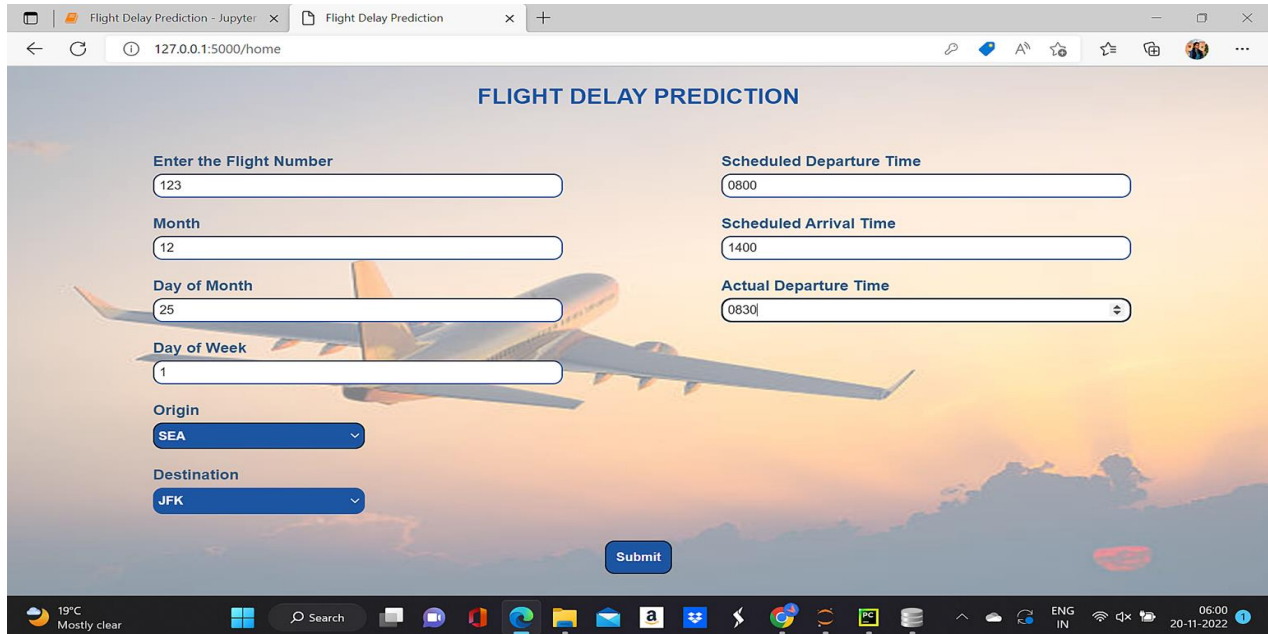


The screenshot shows a web browser window with the title "Flight Delay Prediction - Jupyter" and "ML API". The address bar shows "127.0.0.1:5000/reg". The main content area is titled "REGISTRATION" and contains a form with the following fields:

- Email:
- Password:
- Confirm Password:
- A blue "Register" button.

The browser's taskbar at the bottom shows the date and time as 05:53 on 20-11-2022.

Step 3: After successful login, the user will be directed to the homepage where he/she has to enter all the details of the flight



The screenshot shows a web browser window with the title "Flight Delay Prediction - Jupyter" and "Flight Delay Prediction". The address bar shows "127.0.0.1:5000/home". The page has a background image of an airplane flying over a sunset. The title "FLIGHT DELAY PREDICTION" is centered at the top. Below the title, there are several input fields and dropdown menus for flight details. On the left side, there are fields for "Enter the Flight Number" (123), "Month" (12), "Day of Month" (25), "Day of Week" (1), "Origin" (SEA), and "Destination" (JFK). On the right side, there are fields for "Scheduled Departure Time" (0800), "Scheduled Arrival Time" (1400), and "Actual Departure Time" (0830). A "Submit" button is located at the bottom center. The Windows taskbar is visible at the bottom, showing the time as 06:00 on 20-11-2022.

FLIGHT DELAY PREDICTION

Enter the Flight Number: 123

Month: 12

Day of Month: 25

Day of Week: 1

Origin: SEA

Destination: JFK

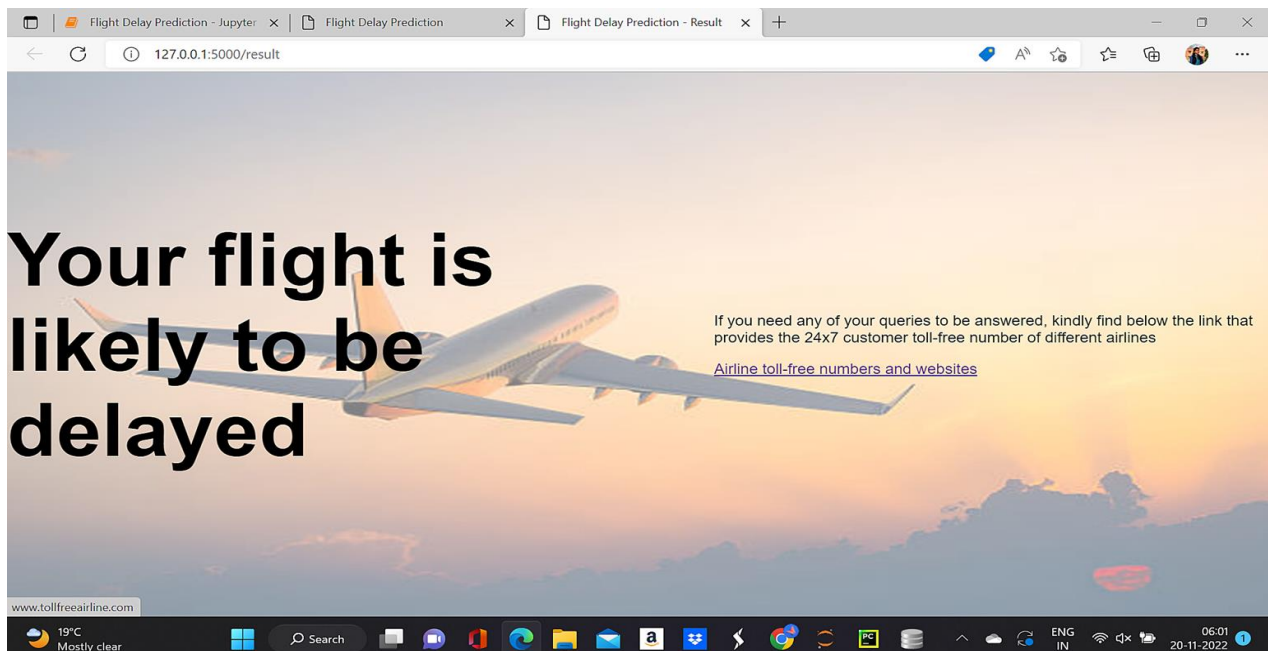
Scheduled Departure Time: 0800

Scheduled Arrival Time: 1400

Actual Departure Time: 0830

Submit

Step 4: After submitting all the flight details, the output of the prediction will be displayed which could either be 'Your flight is likely to be delayed' or 'Your flight is likely to be on time'.



The screenshot shows a web browser window with the title "Flight Delay Prediction - Jupyter" and "Flight Delay Prediction - Result". The address bar shows "127.0.0.1:5000/result". The page has a background image of an airplane flying over a sunset. The text "Your flight is likely to be delayed" is displayed in large, bold, black letters on the left side. On the right side, there is a message: "If you need any of your queries to be answered, kindly find below the link that provides the 24x7 customer toll-free number of different airlines". Below this message is a link: "Airline toll-free numbers and websites". The Windows taskbar is visible at the bottom, showing the time as 06:01 on 20-11-2022.

Your flight is likely to be delayed

If you need any of your queries to be answered, kindly find below the link that provides the 24x7 customer toll-free number of different airlines

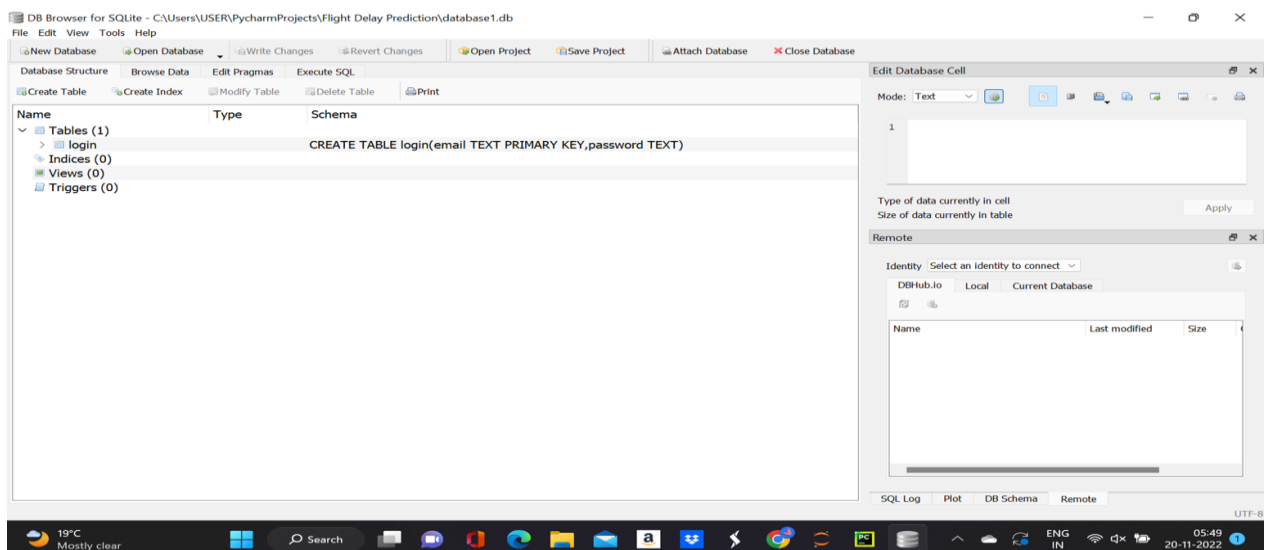
[Airline toll-free numbers and websites](#)

Step 5: To get any of their queries answered by the airlines, the users can click on the link below which provides the toll-free number and websites of the airlines.



7.3 DATABASE SCHEMA

For user registration and login, the SQLite DB Browser is used to store and access data from the database.



8.TESTING AND TEST CASES

8.1 TEST CASES

Test case ID	Feature Type	Component	Test Scenario	Pre-Requlite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Executed By
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	Pravens S
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	Parikhavi T
PredictionPage_TC_OO3	Functional	Home page	Verify user is able to enter the flight details properly	Any Latest 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 Scheduled Arr Time: 1415	User should navigate to result page and input details are recieved properly	Working as expected	Pass	Vaishnavi A

Test case ID	Feature Type	Component	Test Scenario	Pre-Requlite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Executed By
PredictionPage_TC_OO4	Functional	Home Page	Verify user is able to log into application with invalid input	Any Latest 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: MSP Destination: MSP Scheduled Dept Time: 1210 Actual Dept Time: 1231 Scheduled Arr Time: 1415	Application should show 'origin and destination airport can't be same airport' validation message.	Working as expected	Fail	Sally Jemimah J
PredictionPage_TC_OO5	Functional	Home Page	Verify user is able to log into application with invalid input	Any Latest 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: 14 Day of Month: 2 Day of Week: 6 Origin: ATL Destination: MSP Scheduled Dept Time: 1210 Actual Dept Time: 1231 Scheduled Arr Time: 1415	Application should show 'month value can't be more than 12' validation message.	Working as expected	Fail	Vaishnavi A
LoginPage_TC_001	Functional	Login page	Verify if user is able to log into the application with valid username and password	Any Latest Browser	1.Enter URL and click go 2.Enter valid email and password 3.Click the submit button	Email: sallyjemi1908@gmail.com Password: sally	Application should move to homepage where the user can now enter all the flight details	Working as expected	Pass	Sally Jemimah J

Test case ID	Feature Type	Component	Test Scenario	Pre-Requlite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Executed By
LoginPage_TC_002	Functional	Login page	Verify that the user will not be able to log in when entered incorrect email/password	Any Latest Browser	1.Enter URL and click go 2.Enter incorrect email/password 3.Click the submit button	Email: sallyjemi1908@gmail.com Password: hello	Application should show 'incorrect password'	Working as expected	Fail	Sally Jemimah J
RegisterPage_TC_001	Functional	Register page	Verify if the user is able to register with his/her email-id and new password	Any Latest Browser	1.Enter URL and click go 2.Enter the Email-id and password 3.Enter the password once again 4.Click on Register button	Email: leah@gmail.com Password: leah Confirm Password: leah	The login page will be opened as the registration was successful	Working as expected	Pass	Parikhavi T
ResultPage_TC_001	UI	Result page	Verify user is able to view the predicted results	Any Latest Browser	1.Enter URL and click go 2.Enter the correct input values 3.Click the submit button 4.View Results page	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 Scheduled Arr Time: 1415	Application should show 'Flight is on time or Flight is delayed' message.	Working as expected	Pass	Sally Jemimah J

8.2 USER ACCEPTANCE TESTING

1.PURPOSE OF DOCUMENT

The purpose of this document is to briefly explain the test coverage and open issues of the Flight Delay Prediction 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	1	4	2	3	10
Duplicate	0	0	3	0	3
External	0	1	0	0	1
Fixed	1	3	2	6	12
Not Reproduced	0	0	0	0	0
Skipped	1	1	0	0	2
Won't Fix	1	0	0	0	1
Totals	4	9	7	9	29

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
Model Evaluation	10	0	0	10
Client Application	20	0	0	20
Exception Reporting	2	0	0	2
Final Report Output	4	0	0	4

9.RESULTS

9.1 PERFORMANCE METRICS

S.No.	Parameter	Values	Screenshot																														
1.	Metrics	Classification Model: Confusion Matrix - , Accuracy Score- & Classification Report -	<p>Classification Report</p> <pre>print(classification_report(Y_test, Y_pred_log_test))</pre> <table><thead><tr><th></th><th>precision</th><th>recall</th><th>f1-score</th><th>support</th></tr></thead><tbody><tr><td>0.0</td><td>0.96</td><td>0.94</td><td>0.95</td><td>1985</td></tr><tr><td>1.0</td><td>0.60</td><td>0.73</td><td>0.66</td><td>262</td></tr><tr><td>accuracy</td><td></td><td></td><td>0.91</td><td>2247</td></tr><tr><td>macro avg</td><td>0.78</td><td>0.83</td><td>0.81</td><td>2247</td></tr><tr><td>weighted avg</td><td>0.92</td><td>0.91</td><td>0.92</td><td>2247</td></tr></tbody></table> <p>Accuracy, Precision, Recall, F1 Score</p> <pre>acc_log = accuracy_score(Y_test, Y_pred_log_test) prec_log, rec_log, f1_log, sup_log = precision_recall_fscore_support(Y_test, Y_pred_log_test) print('Accuracy Score =', acc_log) print('Precision =', prec_log[0]) print('Recall =', rec_log[0]) print('F1 Score =', f1_log[0])</pre> <p>Accuracy Score = 0.9127725856697819 Precision = 0.9632314862765406 Recall = 0.9378277878885643 F1 Score = 0.9499489274778173</p> <p>Checking for Overfitting and Underfitting</p> <pre>log_train_acc = accuracy_score(Y_train, Y_pred_log_train) log_test_acc = accuracy_score(Y_test, Y_pred_log_test) print('Training Accuracy =', log_train_acc) print('Testing Accuracy =', log_test_acc)</pre> <p>Training Accuracy = 0.9205253784505788 Testing Accuracy = 0.9127725856697819</p>		precision	recall	f1-score	support	0.0	0.96	0.94	0.95	1985	1.0	0.60	0.73	0.66	262	accuracy			0.91	2247	macro avg	0.78	0.83	0.81	2247	weighted avg	0.92	0.91	0.92	2247
	precision	recall	f1-score	support																													
0.0	0.96	0.94	0.95	1985																													
1.0	0.60	0.73	0.66	262																													
accuracy			0.91	2247																													
macro avg	0.78	0.83	0.81	2247																													
weighted avg	0.92	0.91	0.92	2247																													

			<p>Confusion Matrix</p> <pre>pd.crosstab(Y_test.ravel(), Y_pred_log_test)</pre> <table><tr><td>col_0</td><td>0.0</td><td>1.0</td></tr><tr><td>row_0</td><td></td><td></td></tr><tr><td>0.0</td><td>1860</td><td>125</td></tr><tr><td>1.0</td><td>71</td><td>191</td></tr></table>	col_0	0.0	1.0	row_0			0.0	1860	125	1.0	71	191
col_0	0.0	1.0													
row_0															
0.0	1860	125													
1.0	71	191													

2.	Tune the Model	<p><u>Hyperparameter Tuning</u> Validation Method -</p>	<p>Tuning the Hyper Parameters of Logistic Regression</p> <pre>parameters = { 'solver':['newton-cg', 'lbfgs', 'liblinear'], 'C':[100, 10, 1.0, 0.1, 0.01], 'penalty':['l2']}</pre> <pre>In [57]: tuned_model = GridSearchCV(LogisticRegression(max_iter=800), param_grid=parameters, verbose=2) tuned_model.fit(X_train, Y_train.ravel())</pre> <pre>Out[57]: GridSearchCV(estimator=LogisticRegression(max_iter=800), param_grid={'C': [100, 10, 1.0, 0.1, 0.01], 'penalty': ['l2'], 'solver': ['newton-cg', 'lbfgs', 'liblinear']}, verbose=2)</pre> <p>Testing the Tuned Model</p> <pre>: Y_pred_tun_train = tuned_model.predict(X_train) Y_pred_tun_test = tuned_model.predict(X_test)</pre> <pre>: pd.DataFrame(Y_pred_tun_train).value_counts()</pre> <pre>: 0.0 7734 1.0 1250 dtype: int64</pre> <pre>: pd.DataFrame(Y_pred_tun_test).value_counts()</pre> <pre>: 0.0 1922 1.0 325 dtype: int64</pre>
----	----------------	---	---

Evaluating the Tuned Model using Metrics

Classification Report

```
print(classification_report(Y_test, Y_pred_tun_test))
```

	precision	recall	f1-score	support
0.0	0.97	0.94	0.95	1985
1.0	0.61	0.76	0.68	262
accuracy			0.92	2247
macro avg	0.79	0.85	0.81	2247
weighted avg	0.93	0.92	0.92	2247

Accuracy, Precision, Recall, F1 Score

```
acc_tun = accuracy_score(Y_test, Y_pred_tun_test)
prec_tun, rec_tun, f1_tun, sup_tun = precision_recall_fscore_support(Y_test, Y_pred_tun_test)
print('Accuracy Score =', acc_tun)
print('Precision =', prec_tun[0])
print('Recall =', rec_tun[0])
print('F1 Score =', f1_tun[0])
```

Accuracy Score = 0.9158878504672897
Precision = 0.9672210441207075
Recall = 0.9365239204710328
F1 Score = 0.9516252879447147

Checking for Overfitting and Underfitting

```
tun_train_acc = accuracy_score(Y_train, Y_pred_tun_train)
tun_test_acc = accuracy_score(Y_test, Y_pred_tun_test)
print('Training Accuracy =', tun_train_acc)
print('Testing Accuracy =', tun_test_acc)
```

Training Accuracy = 0.9213045414069457
Testing Accuracy = 0.9158878504672897

Confusion Matrix

```
pd.crosstab(Y_test.ravel(), Y_pred_tun_test)
```

col_0	0.0	1.0
row_0		
0.0	1859	126
1.0	63	199

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- With this model, we can easily simplify the extensive traffic at the airport and can prevent the major confusions over flight delays.
- This can enable customer satisfaction and incomes of major airlines.
- Accuracy is measured with the previous models and we have analyzed that this model is much more effective in every way.
- The delay prediction model can make the concerned authorities be well prepared for any possible problem.
- The model can easily be understood by a layman: the model is simple and effective.

DISADVANTAGES

- This model needs to be more compact and flexible. The interoperability feature should be more enhanced.
- The model can be automated instead of manually entering data from the user. Manually entering data is hectic work for the user

11.CONCLUSION

In the present world, the major components of any transportation system include passenger airline, cargo airline and air traffic control system. They all face difficulties due to some sort of miscommunication. Our model has been made with the motive of simplifying complex situations due to flight delays and increasing customer satisfaction. With delays being predicted before, the passengers can easily schedule their plans well before. Our model works with an accuracy of 84% and is considered as an efficient model.

12.FUTURE SCOPE

The project can be extended to a wider range of airports. Current model only supports the data from 5 airports. If the dataset is extended by a vast quantity that has data from airports worldwide then the model can predict any flight delay across the globe. But to do so the complexity of power required will be much greater and the model needs to be trained better to have a higher speed and accuracy of computing results.

13.APPENDIX

SOURCE CODE

register.html

```
<!DOCTYPE html>

<html >

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

  <meta charset="UTF-8">

  <title>ML API</title>

  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet' type='text/css'>
  <link rel="stylesheet" href="{{ url_for('static',filename='styles/Lcssstyle.css') }}">

</head>

<script>

function confirmPass() {

  var pass = document.getElementById("pswd").value

  var confPass = document.getElementById("cpswd").value

  if(pass != confPass) {

    //alert('Wrong confirm password !');

    document.getElementById('error').innerHTML='wrong confirm password';

document.getElementById('rbtn').disabled=true; }
```

```

else
{
    document.getElementById('error').innerHTML="";
document.getElementById('rbtn').disabled=false;
}
}
</script>
<body>
<h1 class="h1">REGISTRATION</h1>
<div class="register">
    <form action="{{ url_for('register')}}" method="post">
        Email: <input type="email" name="email" placeholder="ex:abc@gmail.com" required="required"
autocomplete />
        Password: <input type="password" name="pswd" id="pswd" placeholder="ex:abc@123"
required="required" maxlength=15/>
        Confirm Password:<input type="password" name="cpswd" id="cpswd" placeholder="re-enter same
password" required="required" oninput="confirmPass()" />
        <span id="error" style="color:#F00;"> </span>
        <button type="submit" class="btn btn-primary btn-block" id="rbtn">Register</button>
    </form>
    <br>
    <br>
</div>
</body>
</html>

```

login.html

```
<!DOCTYPE html>

<html>

<!--From https://codepen.io/frytyler/pen/EGdtg-->

<head>

  <meta charset="UTF-8">

  <title>ML API</title>

  <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
  <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
  type='text/css'>
  <link rel="stylesheet" href="{{ url_for('static',filename='styles/Lcssstyle.css') }}">

</head>

<body>

  <h1 class="h1">LOGIN</h1>

  <div class="login">

    <form action="{{ url_for('login') }}" method="post">

      Email: <input type="email" name="email" placeholder="ex:abc@gmail.com" required="required"
      autocomplete />

      Password: <input type="password" name="pswd" id="pswd" placeholder="ex:abc@123"
      required="required" maxlength=15/>

      {{error}}

      <span id="error" style="color:#F00;"> </span>

      <button type="Submit" class="btn btn-primary btn-block" id="lbtn">login</button>

      <h5>If you're a new user, </h5><a href="{{url_for('reg')}}">click here to register</a>

    </form> <br> <br>

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

Flightdelay.html

```
<!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>



<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>

```

result.html

```

<!doctype html>

<html>

<head>

  <title>Flight Delay Prediction - Result</title>

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

</head>

<body>

  {% if prediction[0]== 0.0 %}

  <div class="pred_result" id="result_0">Your flight will likely be on time</div>

  {% endif %}

  {% if prediction[0] == 1.0 %}

  <div class="pred_result" id="result_1">Your flight is likely to be delayed</div>

```



```

    {% endif %}
</body>

<footer>

<p> If you need any of your queries to be answered, kindly find below the link that provides the 24x7
customer toll-free number of different airlines</p>

    <p><a href="http://www.tollfreeairline.com/">Airline toll-free numbers and websites</a>

</p>

</footer>

</html>

```

cssstyle.css

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);

.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-
size: 13px; line-height: 18px; color: #333333; text-align: center; text-shadow: 0 1px 1px rgba(255, 255,
255, 0.75); vertical-align: middle; background-color: #f5f5f5; background-image: -moz-linear-
gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6);
background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6)); background-
image: -webkit-linear-gradient(top, #ffffff, #e6e6e6); background-image: -o-linear-gradient(top,
#ffffff, #e6e6e6); background-image: linear-gradient(top, #ffffff, #e6e6e6); background-repeat:
repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff,
endColorstr=#e6e6e6, GradientType=0); border-color: #e6e6e6 #e6e6e6 #e6e6e6; border-color:
rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -webkit-border-radius:
4px; -moz-border-radius: 4px; border-radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255, 255,
255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px
2px rgba(0, 0, 0, 0.05); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05);
cursor: pointer; *margin-left: .3em; }

.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }

.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-
border-radius: 5px; border-radius: 5px; }

.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position:
0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position
0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s
linear; transition: background-position 0.1s linear; }

```

```
.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }

.btn-primary.active { color: rgba(255, 255, 255, 0.75); }

.btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de, #4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4, GradientType=0); border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }

.btn-primary:hover, .btn-primary.active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }

.btn-block { width: 100%; display:block; }
```

```
* { -webkit-box-sizing:border-box; -moz-box-sizing:border-box; -ms-box-sizing:border-box; -o-box-sizing:border-box; box-sizing:border-box;}
```

```
html { width: 100%; height:150%;}
```

```
body {
```

```
    width: 100%;
```

```
    height:100%;
```

```
    font-family: 'Open Sans', sans-serif;
```

```
    background: #092756;
```

```
    color: #fff;
```

```
    font-size: 16px;
```

```
    text-align:center-left;
```

```
    letter-spacing:1.2px;
```

```
    background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(top, rgba(57,173,219,.25) 0%, rgba(42,60,87,.4) 100%), -moz-linear-gradient(-45deg, #670d10 0%, #092756 100%);
```

```
: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -webkit-linear-gradient(-45deg, #670d10 0%,#092756 100%);
```

```
background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -o-linear-gradient(-45deg, #670d10 0%,#092756 100%);
```

```
background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -ms-linear-gradient(-45deg, #670d10 0%,#092756 100%);
```

```
background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), linear-gradient(to bottom, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), linear-gradient(135deg, #670d10 0%,#092756 100%);
```

```
filter: progid:DXImageTransform.Microsoft.gradient( startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );
```

```
}
```

```
.login {
```

```
position: absolute;
```

```
top: 40%;
```

```
left: 50%;
```

```
margin: -150px 0 0 -150px;
```

```
width:400px;
```

```
height:400px;
```

```
}
```

```
.h1 {font-size: 38px;color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center;margin-left: 90px; }
```

```
body .origin{padding-bottom:10px}
```

```
body .dest{padding-bottom:10px}
```

```
body .btn{padding:20px}
```

```
input {
```

```

width: 100%;
margin-bottom: 10px;
background: rgba(0,0,0,0.3);
border: none;
outline: none;
padding: 10px;
font-size: 13px;
color: #fff;
text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
border: 1px solid rgba(0,0,0,0.3);
border-radius: 4px;
box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
-webkit-transition: box-shadow .5s ease;
-moz-transition: box-shadow .5s ease;
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}

input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2);
}

```

Lcssstyle.css

```

@import url(https://fonts.googleapis.com/css?family=Open+Sans);

.btn { display: inline-block; *display: inline; *zoom: 1; padding: 4px 10px 4px; margin-bottom: 0; font-size: 13px; line-height: 18px; color: #333333; text-align: center; text-shadow: 0 1px 1px rgba(255, 255, 255, 0.75); vertical-align: middle; background-color: #f5f5f5; background-image: -moz-linear-gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-gradient(top, #ffffff, #e6e6e6); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6)); background-image: -webkit-linear-gradient(top, #ffffff, #e6e6e6); background-image: -o-linear-gradient(top,

```

```
#ffffff, #e6e6e6); background-image: linear-gradient(top, #ffffff, #e6e6e6); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff, endColorstr=#e6e6e6, GradientType=0); border-color: #e6e6e6 #e6e6e6 #e6e6e6; border-color: rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -webkit-border-radius: 4px; -moz-border-radius: 4px; border-radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); -moz-box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); cursor: pointer; *margin-left: .3em; }
```

```
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] { background-color: #e6e6e6; }
```

```
.btn-large { padding: 9px 14px; font-size: 15px; line-height: normal; -webkit-border-radius: 5px; -moz-border-radius: 5px; border-radius: 5px; }
```

```
.btn:hover { color: #333333; text-decoration: none; background-color: #e6e6e6; background-position: 0 -15px; -webkit-transition: background-position 0.1s linear; -moz-transition: background-position 0.1s linear; -ms-transition: background-position 0.1s linear; -o-transition: background-position 0.1s linear; transition: background-position 0.1s linear; }
```

```
.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0 rgba(0, 0, 0, 0.25); color: #ffffff; }
```

```
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
```

```
.btn-primary { background-color: #4a77d4; background-image: -moz-linear-gradient(top, #6eb6de, #4a77d4); background-image: -ms-linear-gradient(top, #6eb6de, #4a77d4); background-image: -webkit-gradient(linear, 0 0, 0 100%, from(#6eb6de), to(#4a77d4)); background-image: -webkit-linear-gradient(top, #6eb6de, #4a77d4); background-image: -o-linear-gradient(top, #6eb6de, #4a77d4); background-image: linear-gradient(top, #6eb6de, #4a77d4); background-repeat: repeat-x; filter: progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de, endColorstr=#4a77d4, GradientType=0); border: 1px solid #3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.5); }
```

```
.btn-primary:hover, .btn-primary:active, .btn-primary.active, .btn-primary.disabled, .btn-primary[disabled] { filter: none; background-color: #4a77d4; }
```

```
.btn-block { width: 100%; display: block; }
```

```
* { -webkit-box-sizing: border-box; -moz-box-sizing: border-box; -ms-box-sizing: border-box; -o-box-sizing: border-box; box-sizing: border-box; }
```

```
html { width: 100%; height: 100%; }
```

```
body {
```

```
    width: 100%;
```

```
    height: 100%;
```

```
    font-family: 'Open Sans', sans-serif;
```

```

background: #092756;

color: #ffffff;

font-size: 16px;

text-align:center-left;

letter-spacing:1.2px;

background: -moz-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-gradient(top, rgba(57,173,219,.25) 0%, rgba(42,60,87,.4) 100%), -moz-linear-gradient(-45deg, #670d10 0%, #092756 100%);

background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -webkit-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -webkit-linear-gradient(-45deg, #670d10 0%,#0756 100%);

background: -o-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -o-linear-gradient(-45deg, #670d10 0%,#092756 100%);

background: -ms-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-linear-gradient(top, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), -ms-linear-gradient(-45deg, #670d10 0%,#092756 100%);

background: -webkit-radial-gradient(0% 100%, ellipse cover, rgba(104,128,138,.4) 10%,rgba(192,192,192,0) 40%), linear-gradient(to bottom, rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), linear-gradient(13deg, #67d 0%,#1156 100%);

filter:progid:DXImageTransform.Microsoft.gradient(startColorstr='#c0c0c0',endColorstr='#092756',GradientType=1 );

}

.register {

    position: absolute;

    top: 40%;

    left: 50%;

    margin: -150px 0 0 -150px;

    width:400px;

    height:400px;

}

```

```

.login {
    position: absolute;
    top: 40%;
    left: 50%;
    margin: -150px 0 0 -150px;
    width:400px;
    height:400px;
}

.h1 {font-size: 38px;color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3); letter-spacing:1px; text-align:center;margin-left: 90px;}

body .origin{padding-bottom:10px}
body .dest{padding-bottom:10px}
body .btn{padding:20px}

input {
    width: 100%;
    margin-bottom: 10px;
    background: #ffff;
    border: none;
    outline: none;
    padding: 10px;
    font-size: 13px;
    color: #ffffff;
    text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
    border: 1px solid rgba(0,0,0,0.3);
    border-radius: 4px;
    box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0 1px 1px rgba(255,255,255,0.2);
    -webkit-transition: box-shadow .5s ease;
    -moz-transition: box-shadow .5s ease;
}

```

```
-o-transition: box-shadow .5s ease;
-ms-transition: box-shadow .5s ease;
transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2);
}
```

result_style.css

```
#bgimg {
    position: fixed;
    z-index: -1;
    opacity: 0.5;
    width: 100%;
    height: 100%;
    padding: 0;
    margin: 0;
    top: 0;
}

body {
    font-family: Arial, Helvetica, sans-serif;
    margin: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

div {
```



```
display: flex;
align-items: center;
justify-content: center;
font-size: 82px;
font-weight: 700;
margin: 0;
height: 100vh;}
```

style.css

```
body {
  font-family: Arial, Helvetica, sans-serif;
  margin: 0;
}
.content {
  padding: 10px;
  display: block;
}
.content-head {
  text-align: center;
  font-weight: bold;
  font-size: 36px;
}
.button {
  background-color: #1C55A2;
  color: aliceblue;
  padding: 10px;
  border-radius: 10px;
```

```
border-color: #0E0E0F;
border-width: 1.5px;
}
.button a {
color: aliceblue;
text-decoration: none;
font-weight: bold;
}
#feedback-button {
margin-top: 10px;
}
#feedback-button-section {
text-align: center;
}
#bgimg {
position: fixed;
z-index: -1;
opacity: 0.5;
width: 100%;
height: 100%;
padding: 0;
margin: 0;
top: 0;
}
.centered-head {
text-align: center;
color: #1C55A2;
font-weight: bold;
```

```
}  
.label-item {  
  color: #2E547F;  
  font-weight: bold;  
}  
.detail-container {  
  padding-bottom: 10px;  
  padding-top: 10px;  
}  
.text-input {  
  margin-top: 5px;  
  border-color: #1C55A2;  
  border-width: 1.5px;  
  border-radius: 10px;  
  width: 75%;  
  height: 20px;  
  padding-left: 5px;  
  padding-right: 5px;  
  padding-top: 2px;  
  padding-bottom: 2px;  
}
```

```
.select-input {  
  margin-top: 5px;  
  border-color: #0E0E0F;  
  border-width: 1.5px;  
  border-radius: 10px;
```

```
width: 40%;
height: 30px;
background-color: #1C55A2;
color: aliceblue;
font-weight: bold;
cursor: pointer;
}
#form-content {
  display: flex;
  justify-content: space-evenly;
  flex-direction: row;
  padding-left: 10%;
}
#block1 {
  display: block;
  width: 50%;
  padding: 20px;
}
#block2 {
  display: block;
  width: 50%;
  padding: 20px;
}
#review {
  height: 100px;
  padding-top: 5px;
  font-family: Arial, Helvetica, sans-serif;
}
```

```
#submit-button {
    text-align: center;
    align-items: center;
    display: block;
}

#submit {
    background-color: #1C55A2;
    color: aliceblue;
    font-weight: bold;
}

#submit:hover {
    cursor: pointer;
}

.choose-item {
    font-weight: 600;
}

input[type="radio"], input[type="checkbox"] {
    cursor: pointer;
}

.alert-text {
    color: rgb(255, 79, 47);
    font-size: small;
    padding-left: 10px;
    display: none;
}
```

Importing the required packages

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report, precision_recall_fscore_support
import joblib
import pickle
```

Loading the dataset

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3
```

```
def __iter__(self): return 0
```

```
# @hidden_cell
```

```
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
```

```
# You might want to remove those credentials before you share the notebook.
```

```
cos_client = ibm_boto3.client(service_name='s3',
                              ibm_api_key_id='PM77kN4FyWauT9b7yYQ1Mgxi4SbBI4_3cKxpFreNFXdY',
                              ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
                              config=Config(signature_version='oauth'),
                              endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')
```

```
bucket = 'flightdelayprediction-donotdelete-pr-3uvvg7e82sbqe8r'
```

```
object_key = 'flightdata.csv'
```

```
body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
```

```
# add missing __iter__ method, so pandas accepts body as file-like object
```

```
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(__iter__, body)
```

```
df = pd.read_csv(body)
```

```
df.head()
```

```
df.info()
```

Performing Univariate Analysis

Using Pie Chart

```
df['YEAR'].value_counts().plot(kind='pie', autopct='%0f')  
plt.show()
```

Performing Bivariate Analysis

Using scatterplot

```
plt.scatter(df.DEP_DELAY, df.DEP_DEL15)  
plt.title('Departure Delay Analysis')  
plt.xlabel('DEP_DELAY')  
plt.ylabel('DEP_DEL15')  
plt.show()
```

```
plt.scatter(df.DEP_DELAY, df.DEP_DEL15)  
plt.title('Arrival Delay Analysis')  
plt.xlabel('ARR_DELAY')  
plt.ylabel('ARR_DEL15')  
plt.show()
```

Using lineplots

```
fig, ax = plt.subplots(figsize=(10, 3))  
plt.subplot(1, 2, 1)  
plt.title('CRS_DEP_TIME')  
plt.plot(df.CRS_DEP_TIME)  
plt.subplot(1, 2, 2)  
plt.title('DEP_TIME')  
plt.plot(df.DEP_TIME)  
plt.show()  
fig, ax = plt.subplots(figsize=(10, 3))
```

```
plt.subplot(1, 2, 1)
```

```
plt.title('CRS_ARR_TIME')
```

```
plt.plot(df.CRS_ARR_TIME)
```

```
plt.subplot(1, 2, 2)
```

```
plt.title('ARR_TIME')
```

```
plt.plot(df.ARR_TIME)
```

```
plt.show()
```

```
fig, ax = plt.subplots(figsize=(10, 3))
```

```
plt.subplot(1, 2, 1)
```

```
plt.title('CRS_ELAPSED_TIME')
```

```
plt.plot(df.CRS_ELAPSED_TIME)
```

```
plt.subplot(1, 2, 2)
```

```
plt.title('ACTUAL_ELAPSED_TIME')
```

```
plt.plot(df.ACTUAL_ELAPSED_TIME)
```

```
plt.show()
```

Performing Multivariate Analysis

Using pairplot

```
sb.pairplot(df.iloc[:, 12:])
```

```
plt.show()
```

Using heatmap

```
fig, ax = plt.subplots(figsize=(15, 10))
```

```
sb.heatmap(df.iloc[:, 12:].corr(), annot=True, ax=ax)
```

```
plt.show()
```

Performing Descriptive Analysis

```
df.describe()
```

Dropping unnecessary columns

```
df = df[['FL_NUM', 'MONTH', 'DAY_OF_MONTH', 'DAY_OF_WEEK', 'ORIGIN', 'DEST', 'DEP_DEL15',  
'CRS_ARR_TIME', 'ARR_DEL15']]
```

```
df.head()
```

Handling Missing Values

Checking for null values

```
df.isnull().any()
```


Replacing null values

```
df.fillna(df['DEP_DEL15'].mode()[0], inplace=True)
df.fillna(df['ARR_DEL15'].mode()[0], inplace=True)
```

Checking if the replacement is made

```
df.isnull().any()
```

Handling Outliers

```
fig, ax = plt.subplots(figsize=(5, 6))
sb.boxplot(data=df['CRS_ARR_TIME'])
plt.show()
```

Encoding

One Hot Encoding

```
df = pd.get_dummies(df, columns=['ORIGIN', 'DEST'])
df.head()
df.columns
```

Splitting dataset into Independent and Dependent Variables

```
X = df.drop(columns=['ARR_DEL15'])
Y = df[['ARR_DEL15']]
```

Scaling the Independent Variables

```
scale_crs = StandardScaler()
X[['CRS_ARR_TIME']] = scale_crs.fit_transform(X[['CRS_ARR_TIME']])
scale_flnum = StandardScaler()
X[['FL_NUM']] = scale_flnum.fit_transform(X[['FL_NUM']])
X.head()
X.FL_NUM.value_counts()
```

Converting the Independent and Dependent Variables to 1D Arrays

```
X = X.values
```

```
Y = Y.values
```

Splitting dataset into Train and Test datasets

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2)
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

Building the Machine Learning Models

Support Vector Machine (Classifier)

```
from sklearn.svm import SVC
```

```
svc = SVC()
```

```
svc.fit(X_train, Y_train.ravel())
```

Testing the Models

Support Vector Machine (Classifier)

```
pd.DataFrame(Y_pred_svc_train).value_counts()
```

Classification Report

```
print(classification_report(Y_test, Y_pred_svc_test))
```

Accuracy, Precision, Recall, F1 Score

```
acc_svc = accuracy_score(Y_test, Y_pred_svc_test)
```

```
prec_svc, rec_svc, f1_svc, sup_svc = precision_recall_fscore_support(Y_test, Y_pred_svc_test)
```

```
print('Accuracy Score =', acc_svc)
```

```
print('Precision =', prec_svc[0])
```

```
print('Recall =', rec_svc[0])
```

```
print('F1 Score =', f1_svc[0])
```

Checking for Overfitting and Underfitting

```
vc_train_acc = accuracy_score(Y_train, Y_pred_svc_train)
```

```
svc_test_acc = accuracy_score(Y_test, Y_pred_svc_test)
```

```
print('Training Accuracy =', svc_train_acc)
```

```
print('Testing Accuracy =', svc_test_acc)
```

Confusion Matrix

```
pd.crosstab(Y_test.ravel(), Y_pred_svc_test)
```

Dumping the Chosen Model into pkl file

```
joblib.dump(svc, 'model.pkl')
```

Dumping the Scaling Modules into pkl file

```
pickle.dump(scale_crs, open('crs_scale.pkl', 'wb'))
pickle.dump(scale_flnum, open('flnum_scale.pkl', 'wb'))
```

```
!pip install -U ibm-watson-machine-learning
```

```
from ibm_watson_machine_learning import APIClient
import json
```

```
wml_credentials = {
    "apikey": "G0-fTz5hk2xE6cxkk-PyyzQmio8NNqeMG_wTLTgCSqmU",
    "url": "https://us-south.ml.cloud.ibm.com"
}
```

```
wml_clients = APIClient(wml_credentials)
wml_clients.spaces.list()
space_id = "267b8ab3-7a6f-4efa-81fb-bc602a9f29e2"
wml_clients.set.default_space(space_id)
wml_clients.software_specifications.list(500)
```

```
import sklearn
sklearn.__version__
'1.0.2'
```

```
MODEL_NAME="supportvectormachine"
DEPLOYMENT_NAME="svc_deployment"
DEMO_MODEL=svc
```

```
soft_sepc_id=wml_clients.software_specifications.get_id_by_name("runtime-22.1-py3.9")
```

```
model_props={
    wml_clients.repository.ModelMetaNames.NAME:MODEL_NAME,
    wml_clients.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
    wml_clients.repository.ModelMetaNames.SOFTWARE_SPEC_UID: soft_sepc_id}

```

```
model_details=wml_clients.repository.store_model(model=DEMO_MODEL,meta_props=model_prop
s,training_data=X_train, training_target=Y_train.ravel())
model_details
model_id = wml_clients.repository.get_model_id(model_details)
```

```
model_id
```

```
deployment_props = {  
    wml_clients.deployments.ConfigurationMetaNames.NAME:DEPLOYMENT_NAME,  
    wml_clients.deployments.ConfigurationMetaNames.ONLINE: {}  
}
```

```
deployment = wml_clients.deployments.create(  
    artifact_uid=model_id,  
    meta_props=deployment_props  
)
```

app_ibm.py

```
from flask import Flask, render_template, request, redirect, flash  
import pickle  
import pandas as pd  
import joblib  
import numpy as np  
import sqlite3
```

```
app = Flask(__name__)  
app.secret_key="21433253"  
conn = sqlite3.connect("database1.db")  
conn.execute("CREATE TABLE IF NOT EXISTS login(email TEXT PRIMARY KEY,password TEXT)")  
conn.close()
```

```
@app.route('/')  
def main():  
    return render_template('login.html')
```

```
@app.route('/login', methods=['POST', 'GET'])  
def login():  
    if request.method == 'POST':  
        try:  
            print("request1")  
            fv = [x for x in request.form.values()]  
            print(fv)  
            print([x for x in request.form.values()])  
            print(request.form["email"])
```

```

email = request.form["email"]

pswd = request.form["pswd"]

print("request2")
conn = sqlite3.connect("database1.db")
cur = conn.cursor()
print(email, pswd)
cur.execute("SELECT password FROM login WHERE email=?;", (str(email),))
print("select")

result = cur.fetchone()
cur.execute("SELECT * FROM login")
print(cur.fetchall())
print("fetch")
if result:
    print("You've have been logged in")
    print(result)
    if result[0] == pswd:
        flash("Login successfully", 'success')
        return redirect('/home')
    else:
        return render_template("login.html", error="Please enter correct password")

else:
    print("register")
    flash("Please Register first to access", 'danger')

    return redirect('/reg')

except Exception as e:
    print(e)
    print('danger-----')
    return "hello error"

```

```

@app.route('/reg')
def reg():
    return render_template("register.html")
@app.route('/register', methods=['POST', 'GET'])
def register():
    if request.method == 'POST':
        try:
            print("request1")
            fv = [x for x in request.form.values()]
            print(fv)
            print([x for x in request.form.values()])
            print(request.form["email"])
            email = request.form["email"]
            print(request.form["pswd"])
            pswd = request.form["pswd"]
            conn = sqlite3.connect("database1.db")
            print("database")
            cur = conn.cursor()
            print("cursor")
            cur.execute("SELECT * FROM login WHERE email=?;", (str(email),))
            print("fetch")
            result = cur.fetchone()
            if result:
                print("already")
                flash("User already exists,please login", 'danger')
                return redirect('/')
            else:
                print("insert")
                cur.execute("INSERT INTO login(email,password)values(?,?)", (str(email), str(pswd)))
                conn.commit()
                cur.execute("SELECT * FROM login")
                print(cur.fetchall())
                flash("Registered successfully", 'success')
                return render_template('login.html')

```

```

except Exception as e

    print(e)

    return "hello error1"
@app.route('/home')
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'))
    sdeptime = request.form.get('sdt')
    adeptime = request.form.get('adt')
    arrtime = int(request.form.get('sat'))
    depdelay = int(adeptime) - int(sdeptime)
    inputs = list()
    inputs.append(fl_num)
    inputs.append(month)
    inputs.append(dayofmonth)
    inputs.append(dayofweek)
    if (depdelay < 15):
        inputs.append(0)
    else:
        inputs.append(1)
    inputs.append(arrtime)
    origin = str(request.form.get("org"))
    dest = str(request.form.get("dest"))
    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)

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

def preprocessAndPredict(inputs):
    test_data = np.array(inputs).reshape((1, 16))
    model_file = open('model.pkl', 'rb')
    trained_model = joblib.load(model_file)

    crs_sc = pickle.load(open('crs_scale.pkl', 'rb'))
    flnum_sc = pickle.load(open('flnum_scale.pkl', 'rb'))

```



```

df = pd.DataFrame(data=test_data[0:, 0:],
                  columns=['FL_NUM', 'MONTH', 'DAY_OF_MONTH', 'DAY_OF_WEEK', 'DEP_DEL15',
                           'CRS_ARR_TIME', 'ORIGIN_ATL', 'ORIGIN_DTW', 'ORIGIN_JFK', 'ORIGIN_MSP', 'ORIGIN_SEA',
                           'DEST_ATL', 'DEST_DTW', 'DEST_JFK', 'DEST_MSP', 'DEST_SEA'])
df[['FL_NUM']] = flnum_sc.transform(df[['FL_NUM']])
df[['CRS_ARR_TIME']] = crs_sc.transform(df[['CRS_ARR_TIME']])

data = df.values

result = trained_model.predict(data)

print(result)
return result

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

```

GITHUB LINK:

<https://github.com/IBM-EPBL/IBM-Project-1443-1658388206>

PROJECT DEMO LINK:

<https://youtu.beTOQKYXBQe0I>