UNIVERSITY
GWALIOR • MP • INDIA
"CELEBRATING DREAMS"

TravelSaathi

A Project Report
Submitted in Partial fulfillment of Major Project for the award of Bachelor of Technology in (Computer Science & Engineering)

Submitted to

ITM UNIVERSITY GWALIOR (M.P.)

MAJOR PROJECT REPORT

Submitted by
Atul Anand
[BETN1CS15028]

Under the supervision of

Ms. Geetanjali Surange

Undertaken At

ITM University

Dept. of Computer Science & Engineering

ITM UNIVERSITY

GWALIOR (M.P.)

SESSION-2018-19

UNIVERSITY
GWALIOR • MP • INDIA
"CELEBRATING DREAMS"

CERTIFICATE

This is to certify that Atul Anand, student of B. Tech Final Semester, January 2019– June 2019 sessions of this school has completed his final semester project entitled "TravelSaathi".

He has submitted a satisfactory project report for the award of degree of Bachelor of Technology (B. Tech) of ITM University, Gwalior.

Dr. Sanjay Jain

Associate Professor & Head,

Dept. of Computer Science & Applications,

ITM University, Gwalior

Ms. Geetanjali Surange

Associate Professor,

Dept. of Computer Science & Applications,

ITM University, Gwalior

Acknowledgment

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning of the people where ceaseless cooperation made it possible, whose constant guidance and encouragement crown all the efforts with success.

I shall always be grateful to our parent whose moral support at every step of our life is the reason to us of using our self. Their encouragement in every joy and believe of life laid the real foundation for our personality.

I sincerely thank to my project guide **Ms. Geetanjali Surange** (Asso. Professor, Dept. of Computer Science & Applications, ITM UNIVERSITY) for her great help, guidance, support and ideas towards my project. We are highly grateful to –

Dr. Sanjay Jain (HOD, Dept. of Computer Science & Applications, ITM UNIVERSITY) for providing us their valuable guidance, suggestions, encouragement & ideas during whole project work.

I am highly grateful to my friend **Partha Biswas** for providing me his valuable support with Web Dev & ideas during whole project work.

I am also thankful to staff members for providing us the technical support to carry out the project work and guidance at each & every step during the project work.

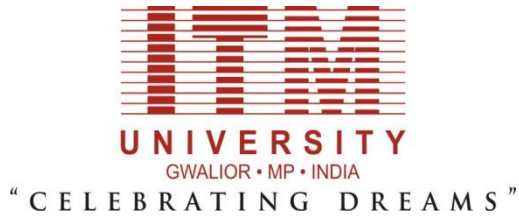
I am also very thankful to lab assistants for their co-operation. It was their careful evaluation, keen interest, timely guidance and valuable suggestions that steered me out of difficulties at every stage of the project. It was great pleasure and good learning experience to have worked under their guidance during tenure of the project.

I also convey my sincere thanks to all my friends who ardently encouraged me through their precious advices and helping attitude in all the fields.

Atul Anand
[BETN1CS15028]

Dept. of Computer Science & Applications

ITM UNIVERSITY, Gwalior (M.P)



DECLARATION

I Atul Anand, student of B. Tech Computer Science and Engineering, Dept. of Computer Science & Engineering, ITM UNIVERSITY, Gwalior(M.P.) hereby declare that the work presented in this Major project is outcome of my own work, is correct to the best of my knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any University for the award of any degree or professional diploma.

Atul Anand
[BETN1CS15028]

Preface

TravelSaathi is an AI enabled Web Platform based application. It serves as the day to day companion for Airport Authority individuals. It can be useful for both aviation professionals and their customers. They can intelligently set the Fare price of flight tickets, can compare with competitive industries, and can also provide environmental friendly and energy saving solutions.

More specifically, this system is designed to allow an Aviation industry to set competitive fare price for their Flight service and obtain beneficiary margin; along with monitoring of Airport premises and conserving Energy.

This web App is cross language compiled software. Its machine learning algorithm and API is written in Python Language and its various Frameworks and Libraries. These days, Python is covering almost all aspects of an IT industry Software production. HTML, CSS, JavaScript has been used for front-end UI interface and back-end connectivity has been handled by PHP language. Flask has been used for API building, and Wamp Server to host WEB APP on local windows PC.

TABLE OF CONTENTS

Chapter 1 Analysis of Proposed System and System requirements

1.1Systems Analysis
1.2 Software Analysis
1.21 Data Dictionary
1.22 Data flow diagram
1.23 Control specification
1.24 Process specification
1.25 State transition diagram
1.26 Object Behavior Diagram
1.27 Operations Attributes
1.28 Technology Stack

Chapter 2 Logical and Physical design of system

2.1Data design
2.2 Architectural design
2.3 Interface design

Chapter 3 Feasibility Study

3.1 Economical Feasibility
3.2 Technical Feasibility
3.3 Behavioral Feasibility

Chapter 4 System Testing and Implementation

4.1Test case design
4.1.1 Black Box Testing
4.1.2 White Box Testing
4.2Test execution
4.2.1 Unit Testing
4.2.2 Integration Testing
4.2.3 System Testing
4.2.4Regression Testing
4.2.5 Acceptance Testing

Chapter 5 Conclusion

Bibliography

Chapter - 1

Analysis of Proposed System and System requirements

1 System Analysis

1.11 Introduction

This document has been written to apply a new version of SRS Software Requirement Specification depends on IEEE – STD - 830 -1998 standard. So, you must compare this document with this standard.
This is the first version for “TravelSaathi”. This document is the basic intended for any individual user, developer, tester, project manager or documentation writer that needs to understand the basic system architecture and its specifications.

1.12 Software Requirement Specification

The purpose of this SRS document is to write the functional and nonfunctional user or system requirements that represent the characteristics of TravelSaathi.

- TravelSaathi is designed to predict optimal Flight Ticket Price.
- Check all competitive Airlines Ticket Fare Price.
- Monitor Heat index of whole Airport Premises.
- One Page multi-solution application.
- Live real data of Domestic Airlines.

1.13 User Requirement Definition

The user requirement for this system is to make the system fast, flexible, less prone to error, reduce expenses and save the time.

- Predict an optimal Fare cost for the Flight Trip.
- Predicted Fare should be competitive to other Aviation companies.
- Get Aware of competitive flight Services.
- Monitor Heat index of Airport Premises.

1.14 The products and process features

- This platform helps Airport authorities to monitor their Airport premises Heat index and live flight Schedules.
- It helps aviation companies to set a cheap and competitive Fare price for the Flight Ticket using regression models.
- They can provide environmental friendly solutions.
- It could also help them to store cold storage or raw food items for longer period.

1.15 System Requirement Specification

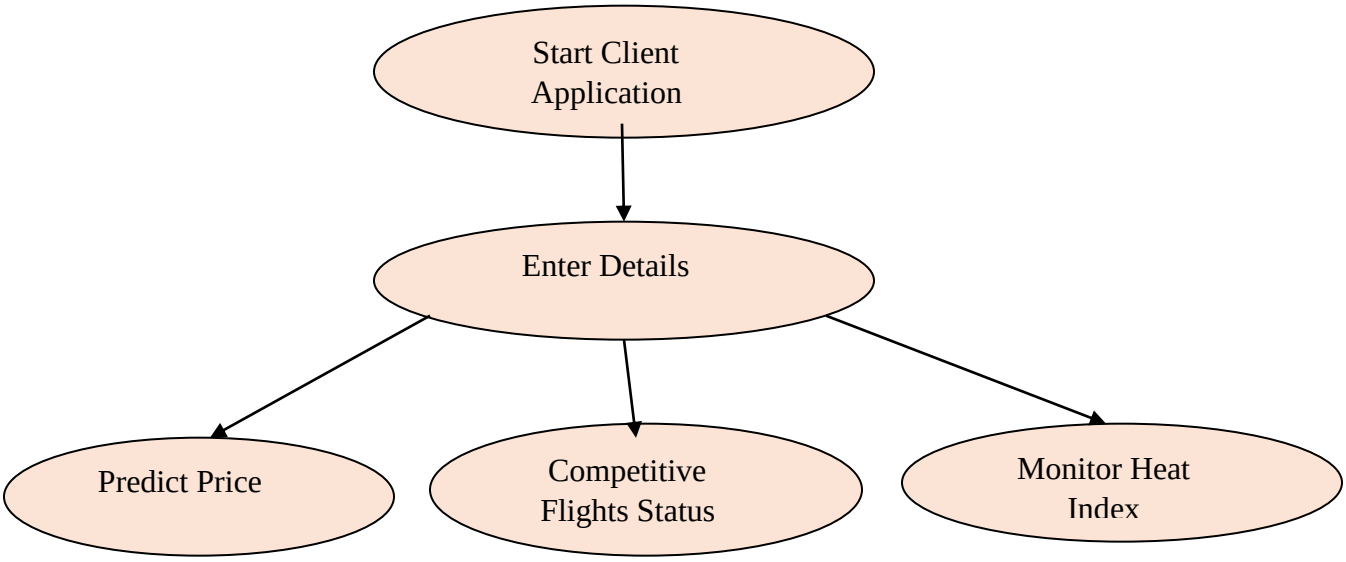
This section gives a functional requirement that is applicable to the Travel companion system.

There are two sub modules in this phase.

- Client side Application
- Server side Application

1.15.1 Client Application

This is a Web platform based UI service. It can easily be fetched through URL link provided. Currently, it works on Local Host Machine. User needs to enter details required on the welcome page. And, a pop – up appears on clicking the “Predict” button. This pop- up holds all Journey related information and also the predicted price. Similarly, real flight schedules and prices can be fetched from UI.

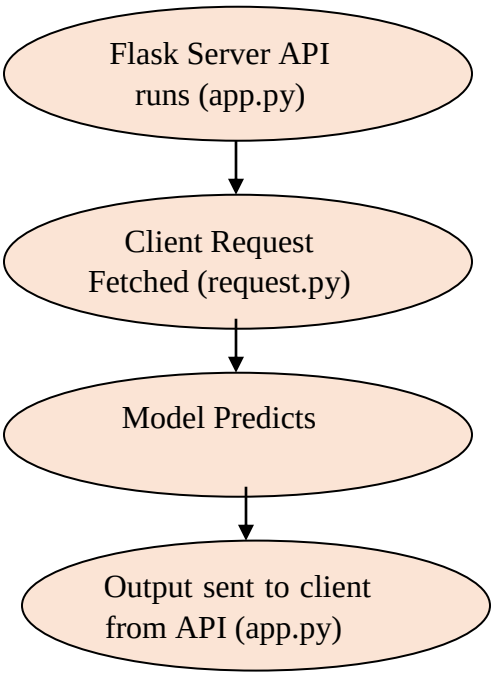


1.15.2 Server Application

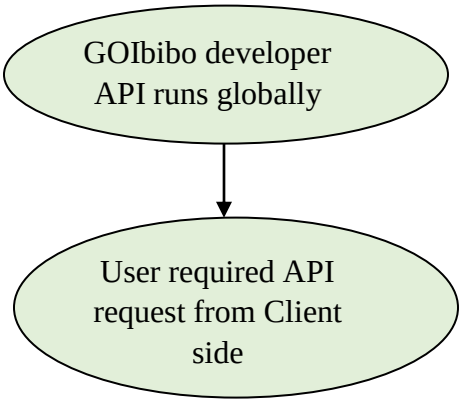
Our application has three different server tasks to achieve all three customer demands. Each of them works in background to fetch user required data on the front end. The Machine learning module is converted to Flask API which gives us the predicted ticket price. The flight status data is fetched from online hosted GoIbibo developer API. And, all of the instances and Databases for the Airport monitoring system has been served and hosted on Google Cloud and Firebase. So, we’ll know about them in three different sections. Viz.

1. Flight Price Prediction server app
2. Flight status Server app.
3. Airport Monitory server app.

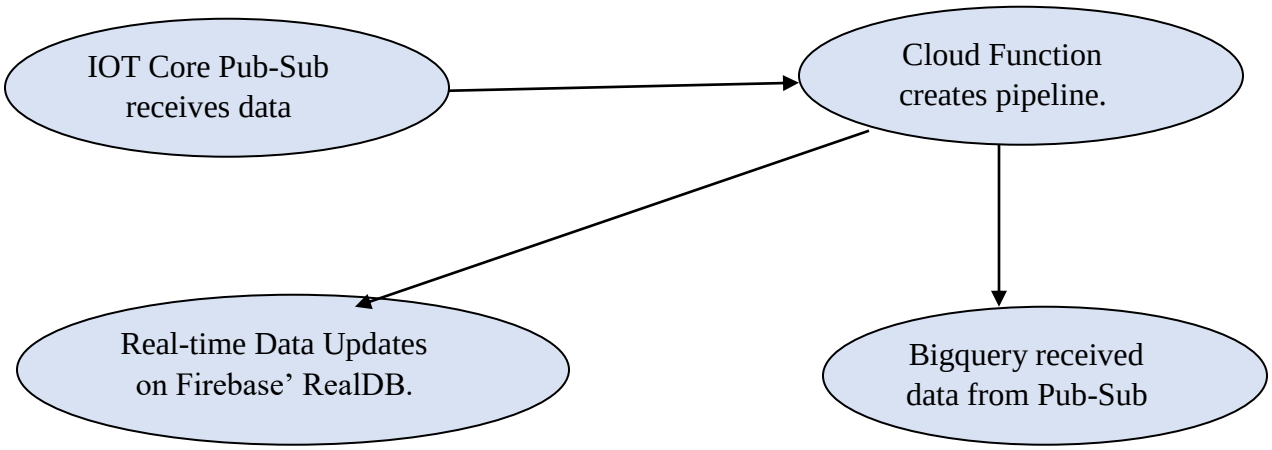
1. **Flight Price Prediction Server app.**



2. **Flight status Server app.**



3. **Airport Monitory server app.**



1.2 Software Analyses

1.21 Data Dictionary

A data dictionary is a file or a set of files that contains a database's metadata. The data dictionary contains records about other objects in the database, such as data ownership, data relationships to other objects, and other data.
The data dictionary is a crucial component of any relational database. Ironically, because of its importance, it is invisible to most database users. Typically, only database administrators interact with the data dictionary.

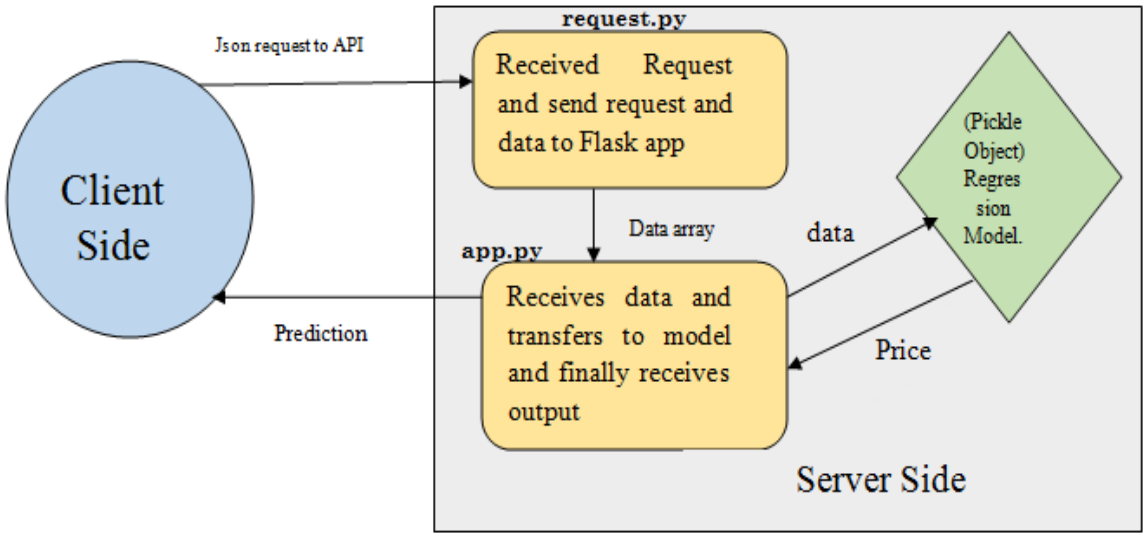
1.22 Data Flow Diagram

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system) The DFD also provides information about the outputs and inputs of each entity and the process itself.

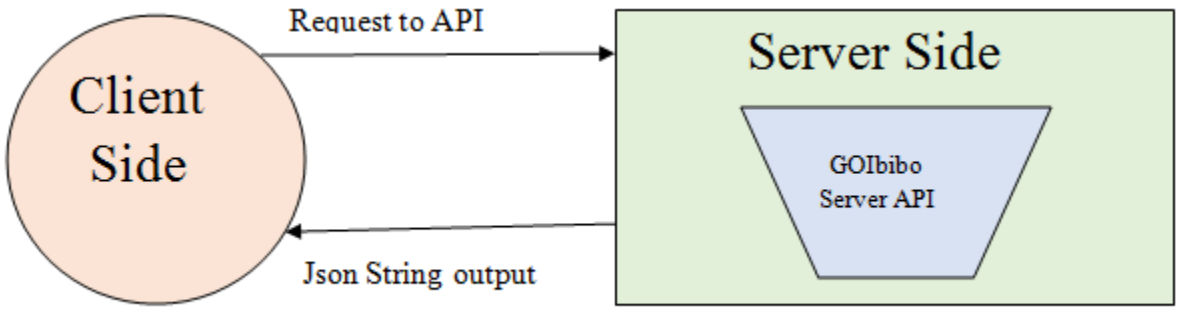
We need to represent 3 separate DFDs to show all three tasks in our project. Viz.

1. Flight Price
2. Flight Status
3. Airport Premises Monitor

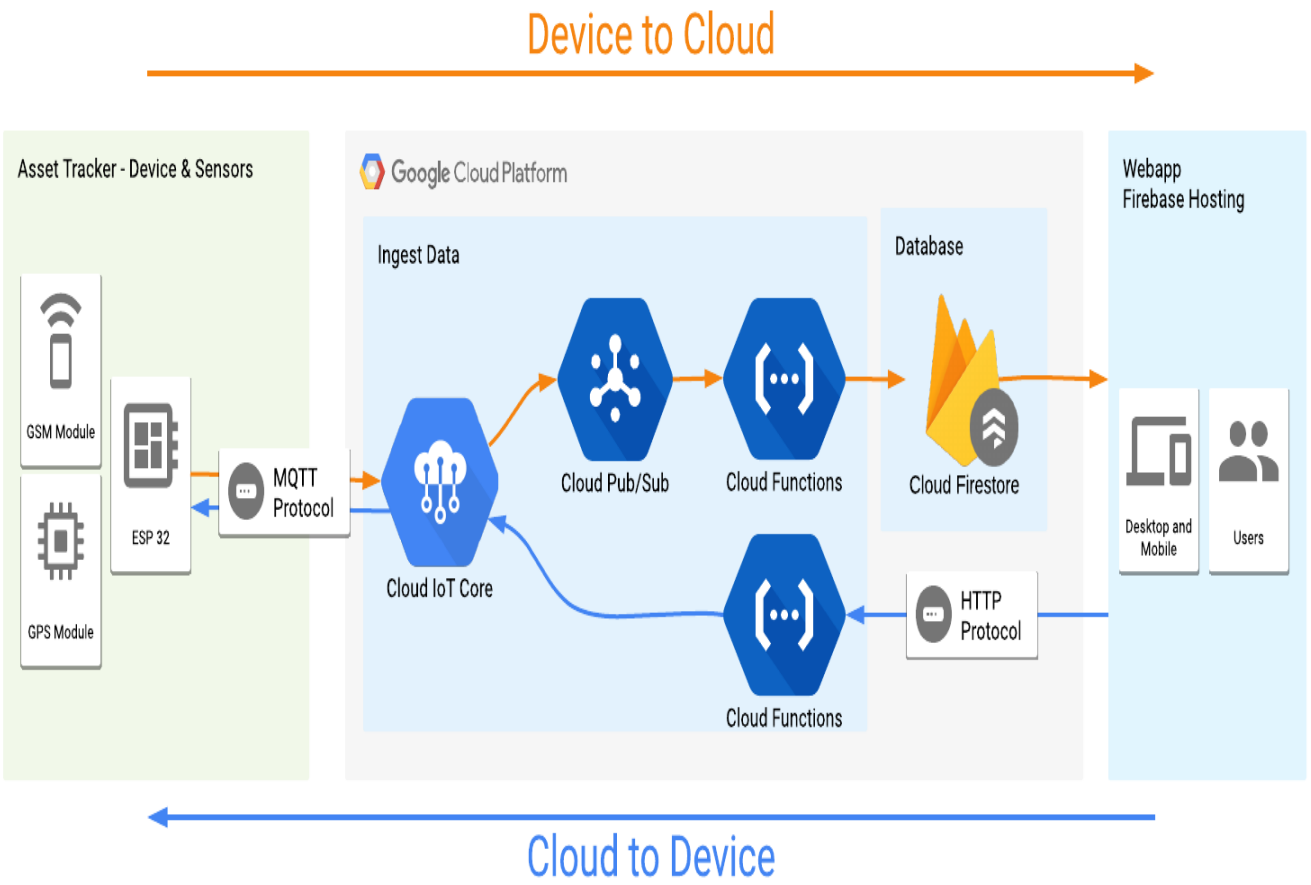
- **Flight Price :**



- **Flight Status :**

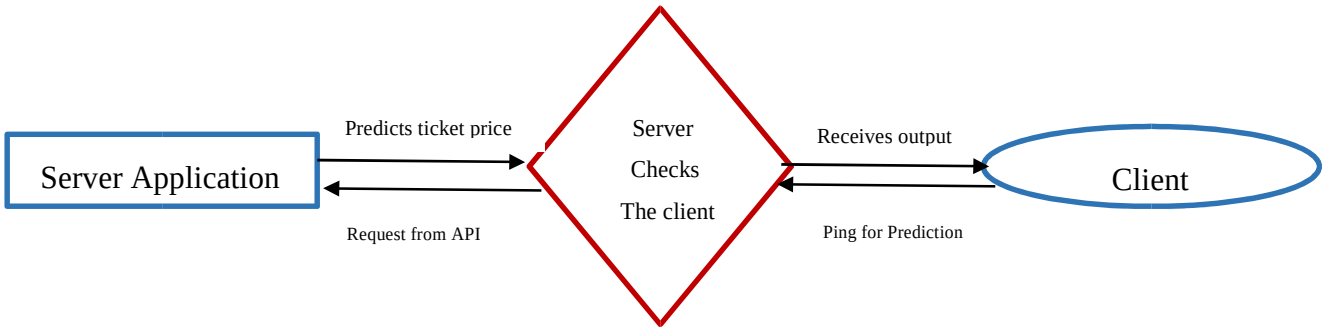


- **Airport Premises Monitor :**



1.23 Control Flow Diagram

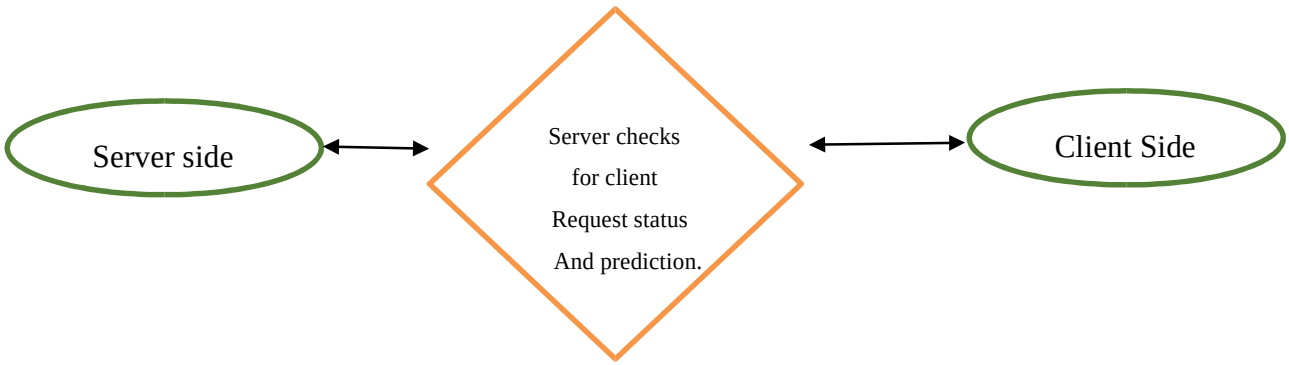
A control flow diagram is a diagram to describe the control flow of a business process, process or review. Control flow diagrams were developed in the 1950s, and are widely used in multiple engineering disciplines



1.24 Process Specification

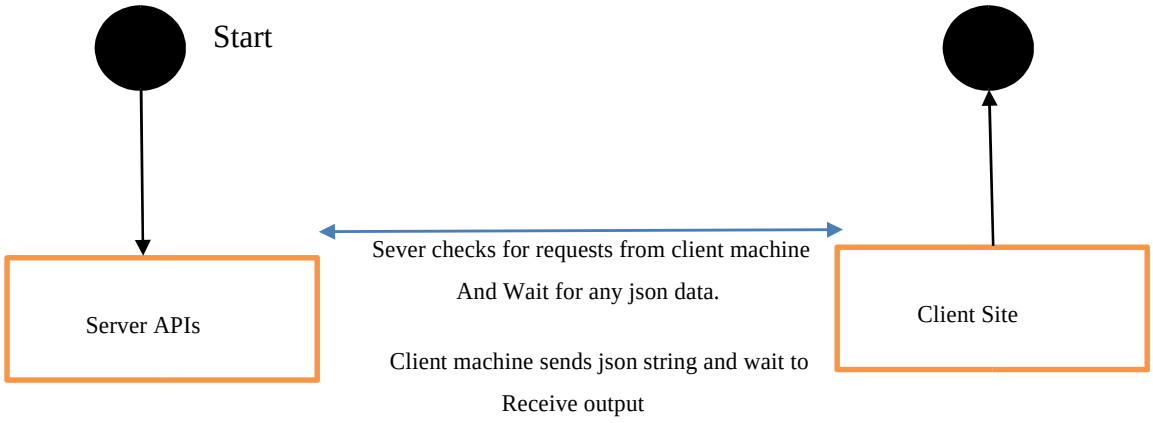
A process specification is a method used to document, analyze and explain the decision-making logic and formulas used to create output data from process input data. Its objective is to flow down and specify regulatory/engineering requirements and procedures. High-quality, consistent data requires clear and complete process specifications.

A process specification reduces ambiguity, allowing an individual or organization to obtain a precise description of executed tasks and accomplishments and validate system design, including the data dictionary and data flow diagrams.



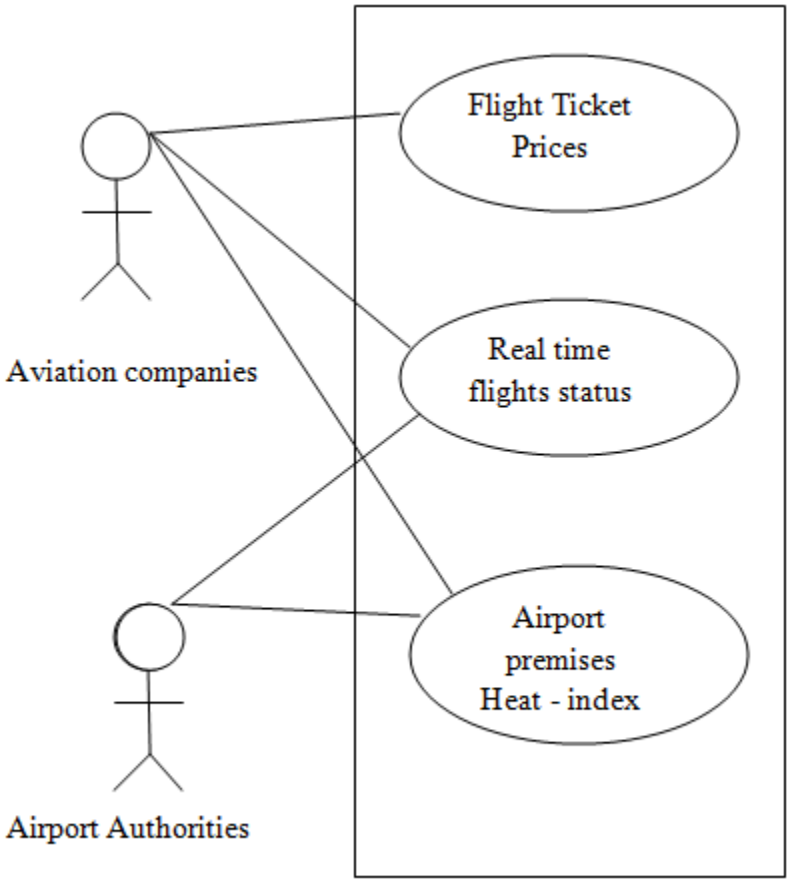
1.25 State Transition Diagram

A state diagram is used to represent the condition of the system or part of the system at finite instances of time. It's a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams



1.26 Object Behavior Diagram

UML behavioral diagrams visualize, specify, construct, and document the dynamic aspects of a system. The behavioral diagrams are categorized as follows: use case diagrams, interaction diagrams, state–chart diagrams, and activity diagrams.



1.27 Operations, Attribute

- Flask server API hosts the machine learning Model packed as pickle object and message passed in json string format.
- Cloud provides IOT core and Pub-Sub and the entire message is transferred using MQTT protocol. Cloud Function provides pipeline for data transfer.
- BigQuery holds all the historical data and real time updates are reflected on Firebase.
- Regression models are used for ticket price prediction. Data received as json string and output produced as a string object.

1.28 Technology Stack

This is a cross language compiled application. The software is written in various languages. Viz. python, JavaScript, PHP, etc. and different libraries are used for serving APIs; viz. Flask, urlib, request, IOT Core, Pub-Sub, etc. Lets study them in detail.

1. Flask API
2. GoIbibo API
3. Firebase
4. Wamp Server
5. IoT Core – PubSub
6. BigQuery
7. Python 3.5.6
8. JavaScript
9. PHP

1.28.1 Flask API

Flask is a web framework for Python, meaning that it provides functionality for building web applications, including managing HTTP requests and rendering templates. In this section, we will create a basic Flask application. In later sections, we'll add to this application to create our API. Don't worry if you don't understand each individual line of code yet—explanations will be forthcoming once you have this initial version of the application working.

What is an API?

A web API allows for information or functionality to be manipulated by other programs via the internet. For example, with Twitter's web API, you can write a program in a language like Python or Javascript that can perform tasks such as favorite tweets or collecting tweet metadata.

In programming more generally, the term API, short for Application Programming Interface, refers to a part of a computer program designed to be used or manipulated by another program, as opposed to an interface designed to be used or manipulated by a human. Computer programs frequently need to communicate amongst them or with the underlying operating system, and APIs are one way they do it. In this tutorial, however, we'll be using the term API to refer specifically to web APIs.

API Terminology

When using or building APIs, you will encounter these terms frequently:

- **HTTP (Hypertext Transfer Protocol):** is the primary means of communicating data on the web. HTTP implements a number of "methods," which tell which direction data is moving and

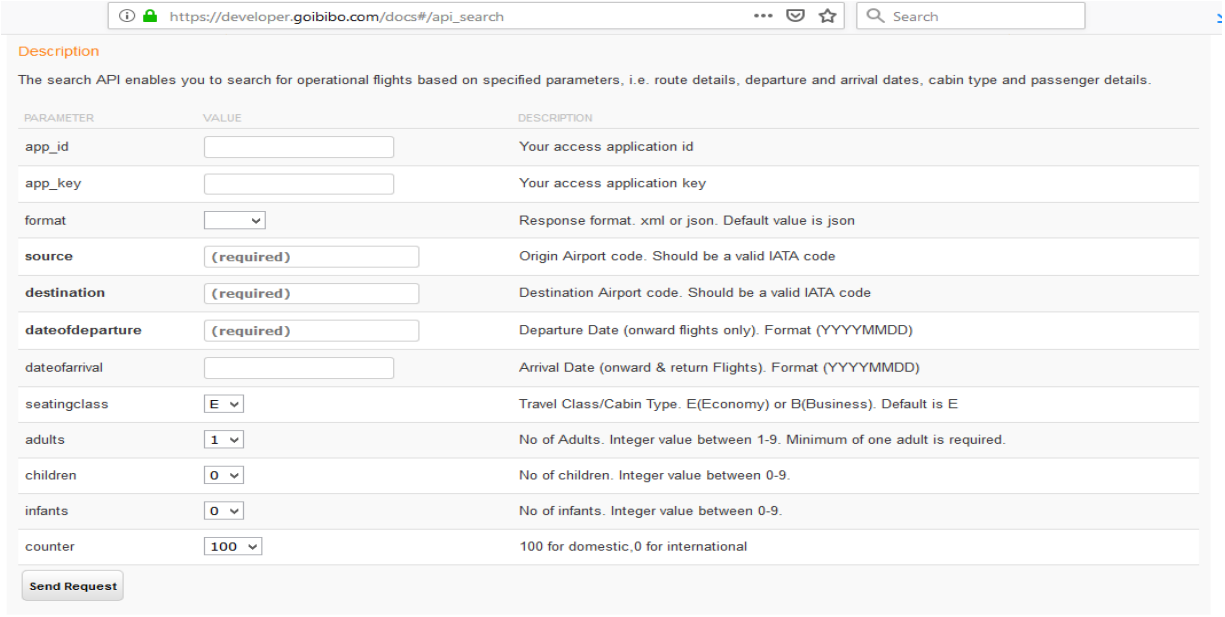
what should happen to it. The two most common are GET, which pulls data from a server, and POST, which pushes new data to a server.

- **URL (Uniform Resource Locator)**: An address for a resource on the web, such as *https://programminghistorian.org/about*. A URL consists of a protocol (http://), domain (*programming*historian.org), and optional path (/about). A URL describes the location of a specific resource, such as a web page. When reading about APIs, you may see the terms URL, request, URI, or endpoint used to describe adjacent ideas. This tutorial will prefer the terms URL and request to avoid complication. You can follow a URL or make a GET request in your browser, so you won't need any special software to make requests in this tutorial.
- **JSON (JavaScript Object Notation):** is a text-based data storage format that is designed to be easy to read for both humans and machines. JSON is generally the most common format for returning data through an API, XML being the second most common.
- **REST (Representational State Transfer):** is a philosophy that describes some best practices for implementing APIs. APIs designed with some or all of these principles in mind are called REST APIs. While the API outlined in this lesson uses some REST principles, there is a great deal of disagreement around this term. For this reason, I do not describe the example APIs here as REST APIs, but instead as web or HTTP APIs.

1.28.2 GOIbibo API

Domain for all APIs is 'developer.goibibo.com'. And these contain search and data APIs only, if you would like to integrate our booking flow and payment gateway, please contact them.

Flight Search API: The search API enables you to search for operational flights based on specified parameters, i.e. route details, departure and arrival dates, cabin type and passenger details.



1.28.3 Firebase

Firebase is a mobile and web application development platform developed by Firebase, Inc. in 2011, then acquired by Google in 2014. As of October 2018, the Firebase platform has 18 products, which are used by 1.5 million apps.

Firebase is a Backend-as-a-Service—**BaaS**—that started as a YC11 startup and grew up into a next-generation app-development platform on Google Cloud Platform.

Firebase Services

1.28.4 Wamp Server

WampServer is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySql database. Alongside, Phpmyadmin allows you to manage easily your databases.

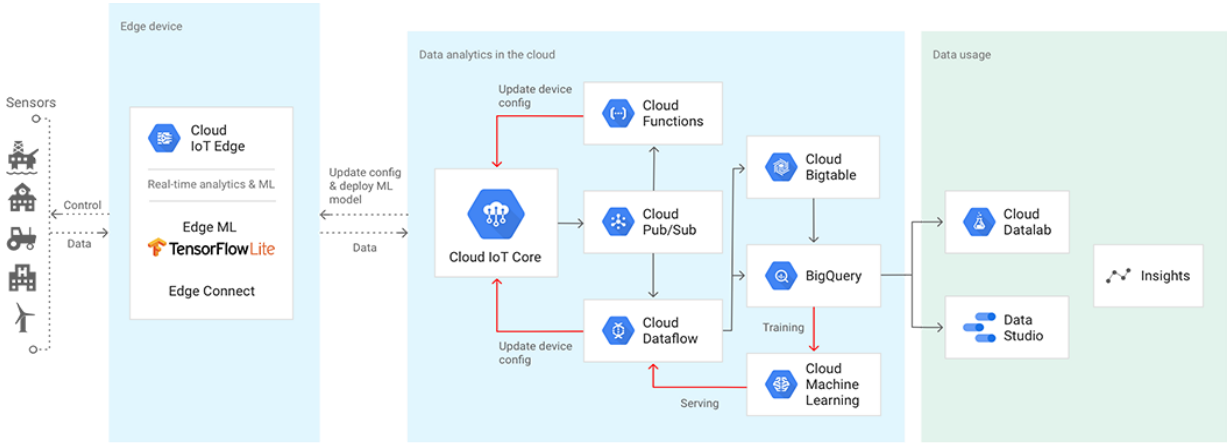
WampServer refers to a software stack for the Microsoft Windows operating system, created by Romain Bourdon and consisting of the Apache web server, OpenSSL for SSL support, MySql database and PHP programming language.

1.28.5 IoT Core – PubSub

Cloud IoT Core

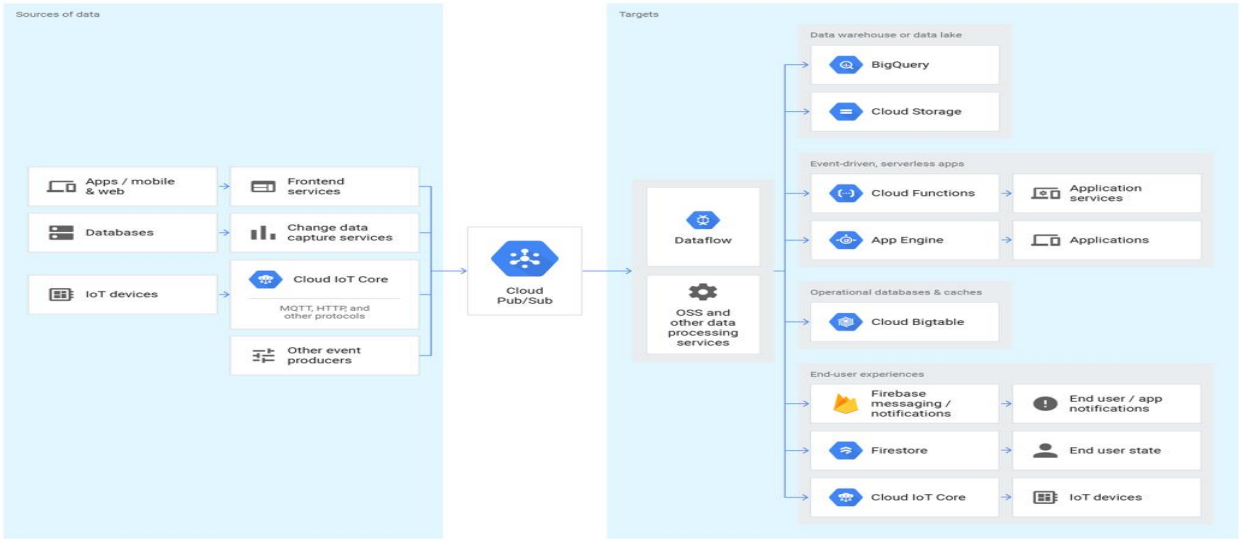
A fully managed service to easily and securely connect, manages, and ingests data from globally dispersed devices.

Cloud IoT Core is a fully managed service that allows you to easily and securely connect, manage, and ingest data from millions of globally dispersed devices. Cloud IoT Core, in combination with other services on Cloud IoT platform, provides a complete solution for collecting, processing, analyzing, and visualizing IoT data in real time to support improved operational efficiency.



Cloud Pub/Sub

Cloud **Pub**/**Sub** is a scalable, durable, event ingestion and delivery system that supports publish - subscribe pattern at large and small scales. Cloud **Pub**/**Sub** makes your systems more robust by decoupling publishers and subscribers of event data.

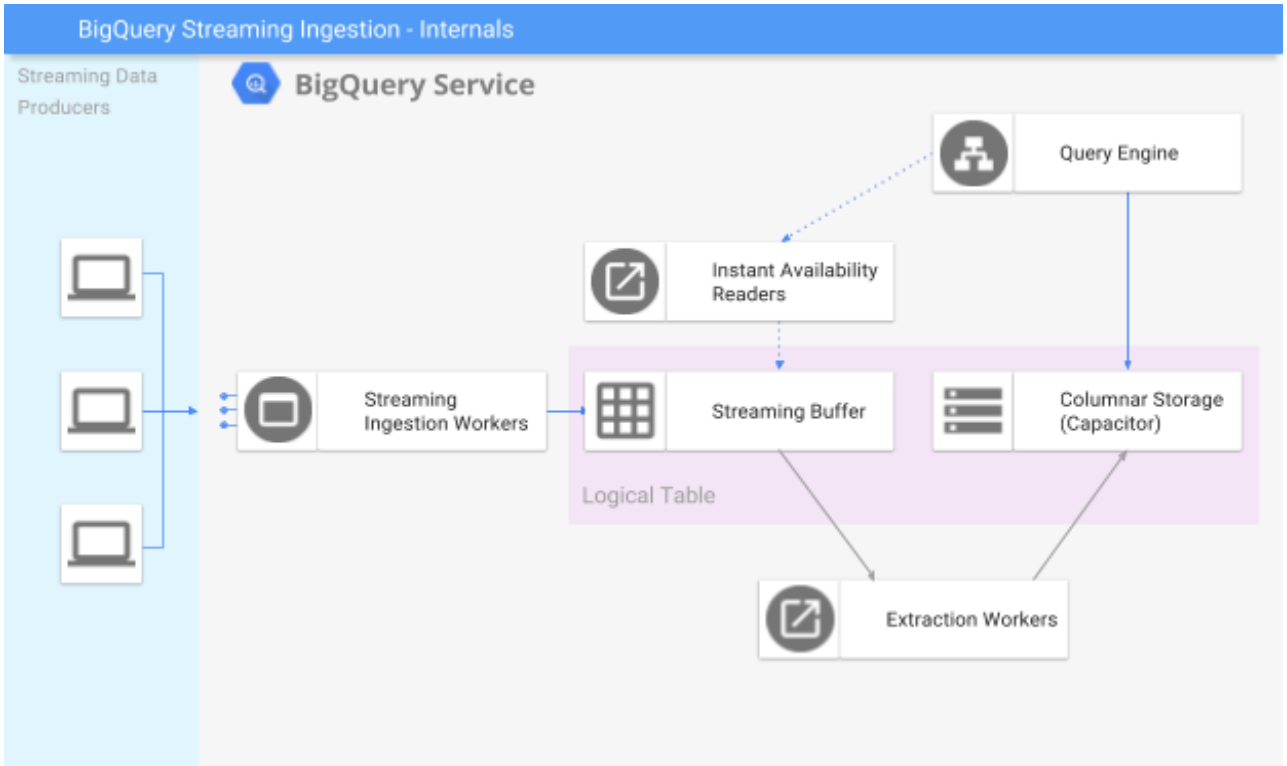


1.28.6 BigQuery

BigQuery is a server less, highly-scalable, and cost-effective cloud data warehouse with an in-memory BI Engine and machine learning built in.

BigQuery, Google's server less, highly scalable enterprise data warehouse, is designed to make data analysts more productive with unmatched price-performance. Because there is no infrastructure to manage, you can focus on uncovering meaningful insights using familiar SQL without the need for a database administrator.

Analyze all your batch and streaming data by creating a logical data warehouse over managed columnar storage, as well as data from object storage and spreadsheets. Create blazing-fast dashboards and reports with the in-memory BI Engine. Build and operationalize machine learning solutions or carry out geospatial analysis using simple SQL. Securely share insights within your organization and beyond as datasets, queries, spreadsheets, and reports. BigQuery's powerful streaming ingestion captures and analyzes data in real time, ensuring insights are always current. Plus, you can analyze up to 1 TB of data and store 10 GB of data for free each month.

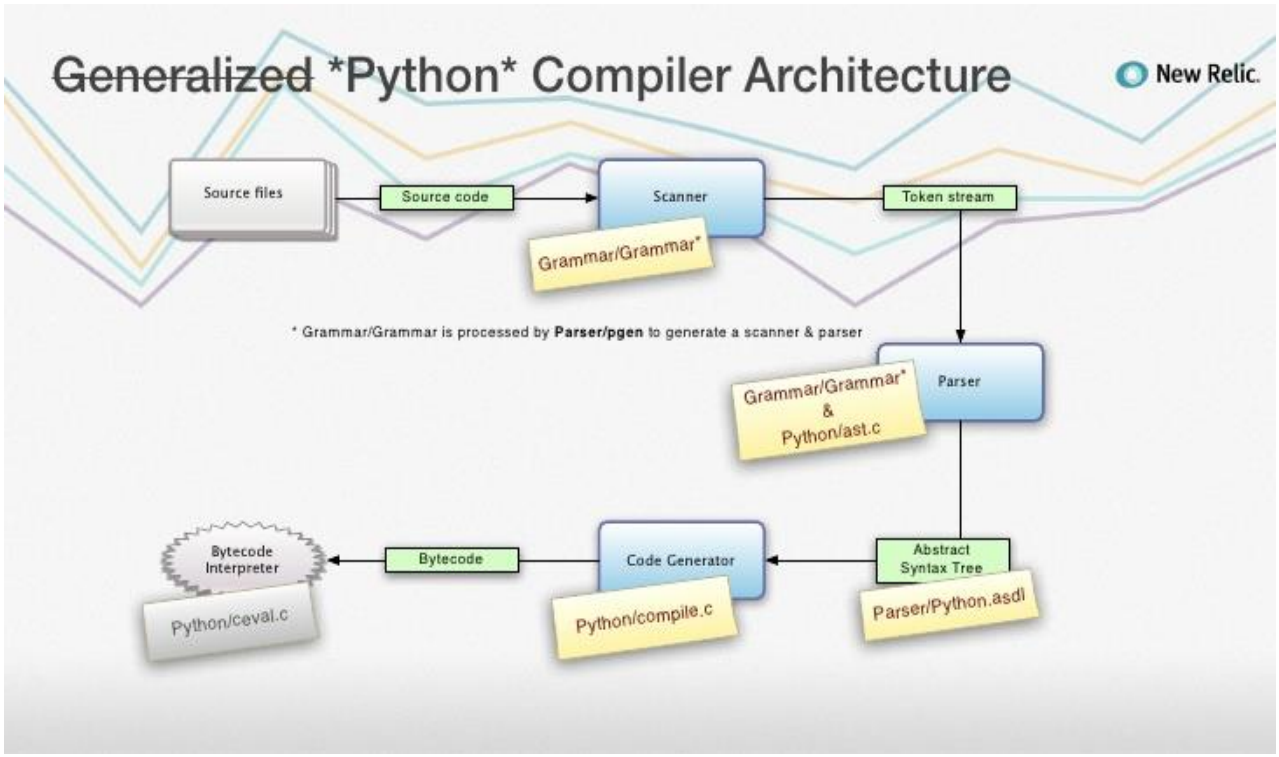


1.28.7 Python 3.5.6

Python is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that

emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.

We have hosted our **Flask API** script using **python scripting language version 3.5.6** .



1.28.8 JavaScript

JavaScript is a prototype-based, multi-paradigm, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

JavaScript, often abbreviated as JS, is a high-level, interpreted programming language that conforms to the ECMA Script specification. JavaScript has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

We have implemented our **Cloud Function data Pipeline** using **JavaScript** language.

1.28.9 PHP

PHP: Hypertext Preprocessor is a general-purpose programming language originally designed for web development. It was originally created by Rasmus Lerdorf in 1994; the PHP reference implementation is now produced by The PHP Group.

We have written **backend** of our whole **Website** using **PHP** language.

Chapter 2

Logical and Physical design of the System

2.1 Data Design

Data design is the first design activity, which results in fewer complexes, modular and efficient program structure. The information domain model developed during analysis phase is transformed into data structures needed for implementing the software. The data objects, attributes, and relationships depicted in entity relationship diagrams and the information stored in data dictionary provide a base for data design activity. During the data design process, data types are specified along with the integrity rules required for the data.

For specifying and designing efficient data structures, some principles should be followed. These principles are listed below.

1. The data structures needed for implementing the software as well-as the operations that can be applied on them should be identified.

2. A data dictionary should be developed to depict how different data objects interact with each other and what constraints are to be imposed on the elements of data structure.

3. Stepwise refinement should be used in data design process and detailed design decisions should be made later in the process.

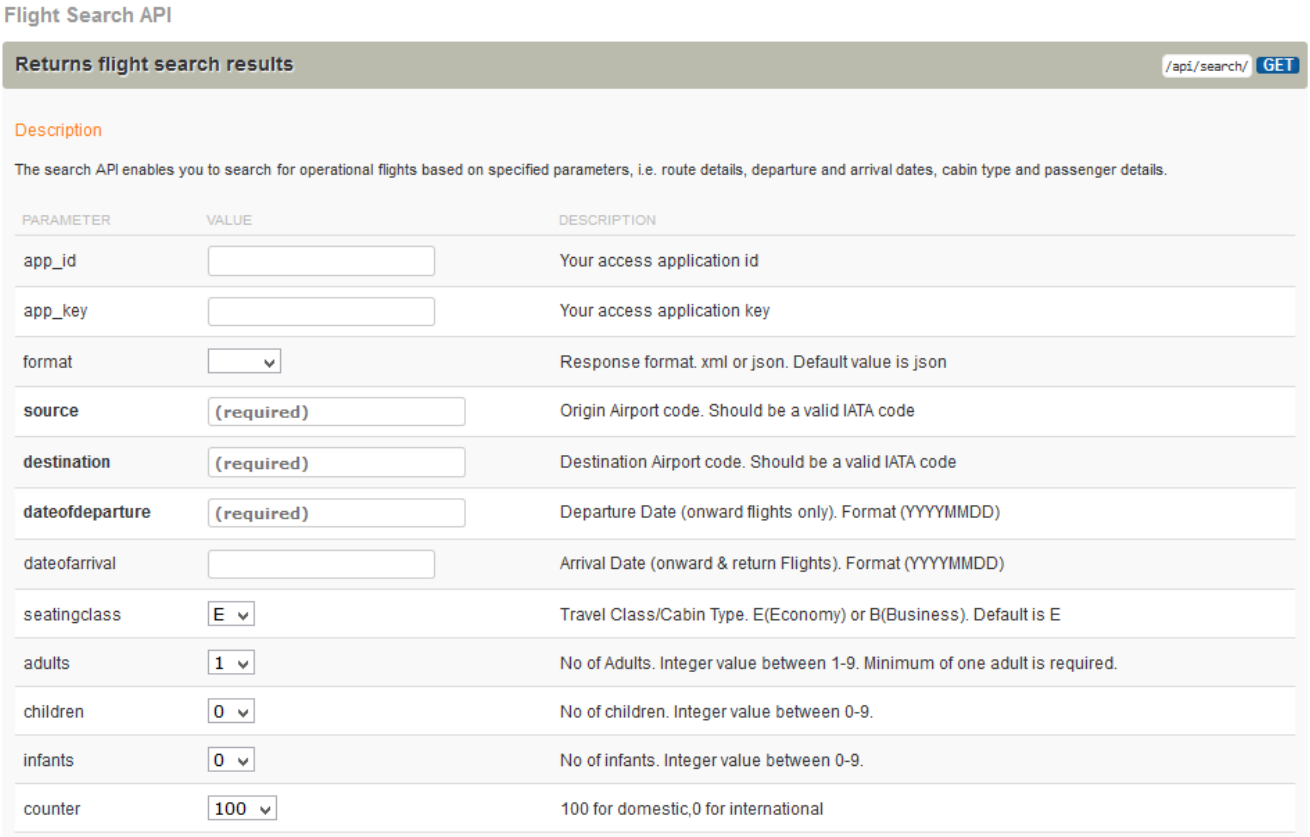
4. Only those modules that need to access data stored in a data structure directly should be aware of the representation of the data structure.

5. A library containing the set of useful data structures along with the operations that can be performed on them should be maintained.

6. Language used for developing the system should support abstract data types.

- **Data/ Object Description:**

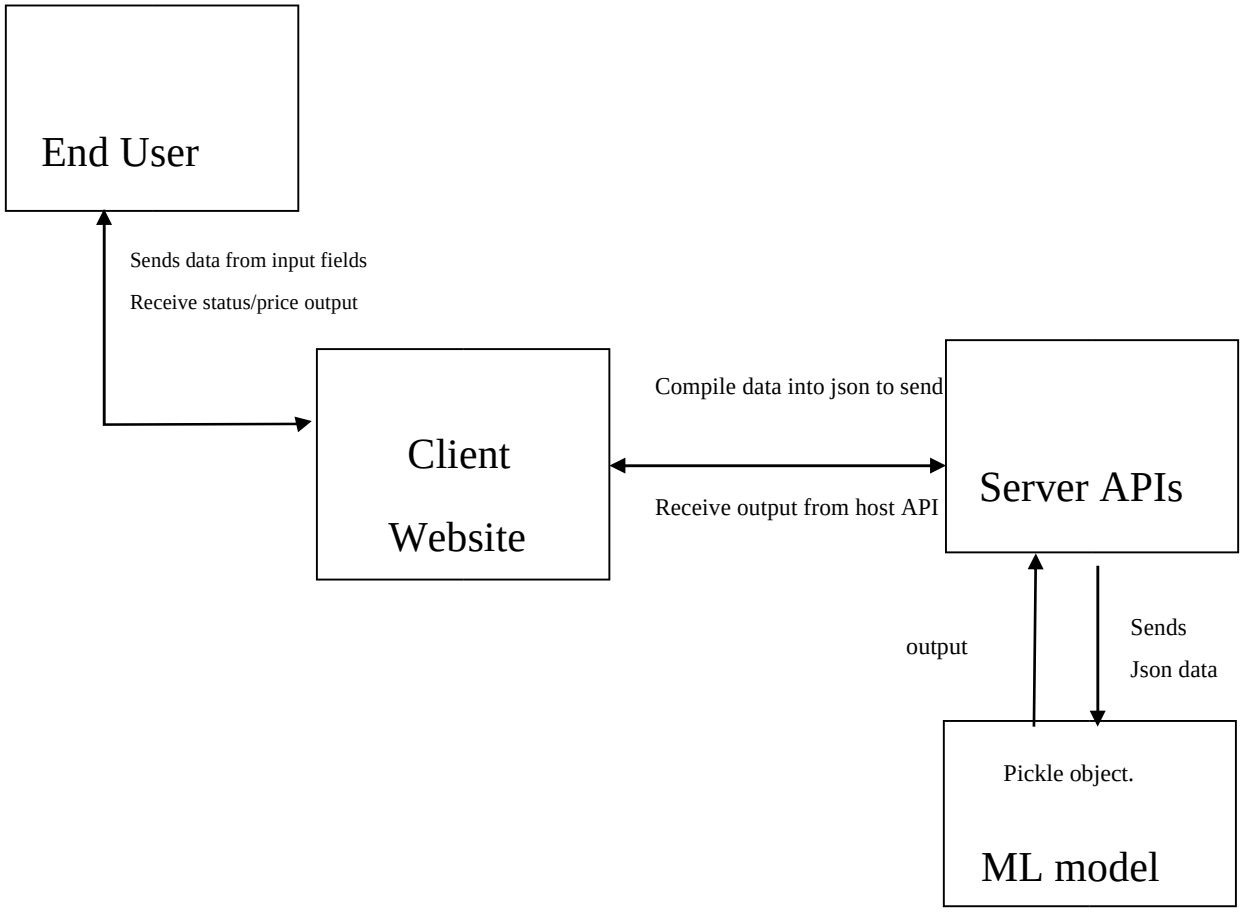
    - There were total 12 attributes to the data dictionary table, which we imported from goIbibo developer API.
    - We imported total of 7 attributes into our application. Viz. ***Source, Destination, Duration, Airlines, Date of Journey, stops, Fare***.



2.2 Architectural Design

The software needs the architectural design to represents the design of software. IEEE defines architectural design as "the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system." The software that is built for computer-based systems can exhibit one of these many architectural styles. Each style will describe a system category that consists of:

- A set of components (e.g.: a database, computational modules) that will perform a function required by the system.
- The set of connectors will help in coordination, communication, and cooperation between the components.
- Conditions that how components can be integrated to form the system.
- Semantic models that help the designer to understand the overall properties of the system



ML Model:

→ The ML model is an emsembled regression model consisting of Random Forest as base model, and supportive models; like: XGBoost, LightGBM, Elastnic Net, Random Forest, Lasso Regression.
→ Total of 38 features were generated from 10 given features; out of which 25 were chosen for model building.

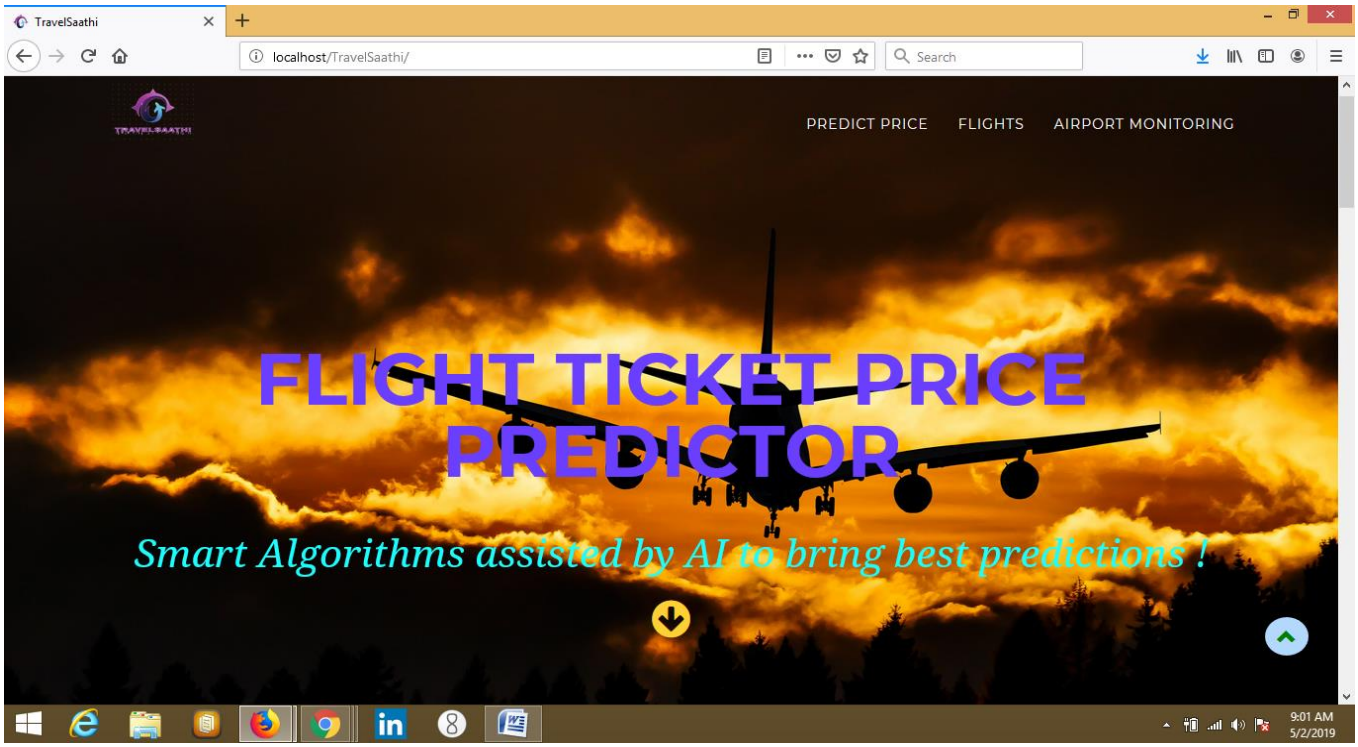
2.3 Interface Design

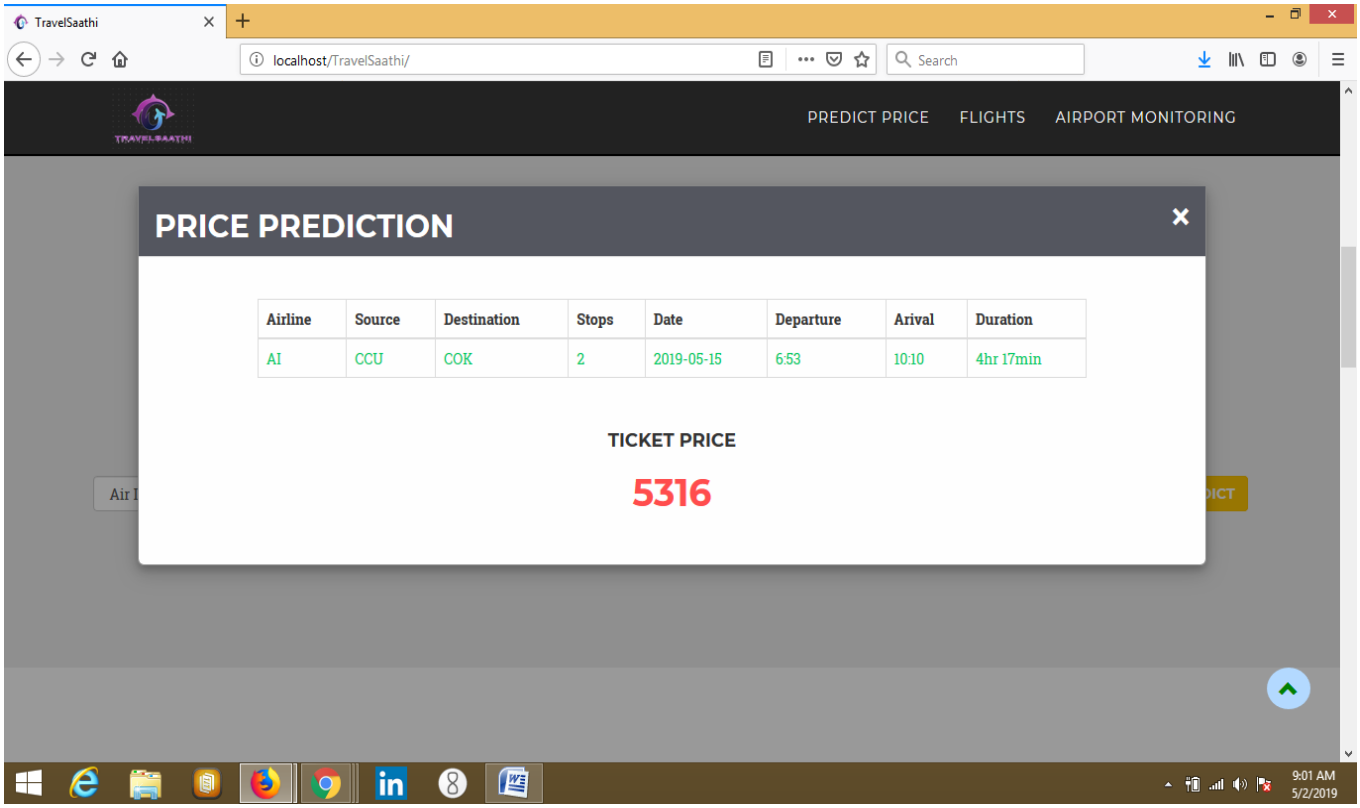
User interface (UI) design is the process of making interfaces in software or computerized devices with a focus on looks or style. Designers aim to create designs users will find easy to use and pleasurable. UI design typically refers to graphical user interfaces but also includes others, such as voice-controlled ones.

User interface design requires a good understanding of user needs. There are several phases and processes in the user interface design, some of which are more demanded upon than others, depending on the project. (Note: for the remainder of this section, the word system is used to denote any project whether it is a website, application, or device.)

- **Graphical user interface design** – actual look and feel design of the final graphical user interface (GUI). These are design's control panels and faces; voice

controlled interfaces involve oral-auditory interaction, while gesture-based interfaces witness users engaging with 3D design spaces via bodily motions. It may be based on the findings developed during the user research, and refined to fix any usability problems found through the results of testing.





Chapter 3

Feasibility Study

3 Feasibility Study

3.1 ECONOMIC FEASIBILITY

Economic analysis is most frequently used for evaluation of the effectiveness of the system. More commonly known as cost/benefit analysis the procedure is to determine the benefit and saving that are expected from a system and compare them with costs, decisions is made to design and implement the system.

This part of feasibility study gives the top management the economic justification for the new system. This is an important input to the management the management, because very often the top management does not like to get confounded by the various technicalities that bound to be associated with a project of this kind. A simple economic analysis that gives the actual comparison of costs and benefits is much more meaningful in such cases.

In the system, the organization is most satisfied by economic feasibility. Because, if the organization implements this system; it need not require any additional hardware resources as well as it will be saving lot of time.

- **Developer's Perspective to Economic Feasibility:**

    - The overall cost of building this software is not too high.
    - This made the production and market cost of this Software feasible.
    - Different Cost factors to this software are:
        - Developer's Wage
        - Nominal infrastructure (PC, Internet, Electricity, etc)
        - Google Cloud's Monthly rental (~ Rs. 1000 / month)
        - Hardware sensors (1000 each)

3.2 TECHNICAL FEASIBILITY

Technical feasibility centers on the existing manual system of the test management process and to what extent it can support the system. According to feasibility analysis procedure the technical feasibility of the system is analyzed and the technical requirements such as software facilities, procedure, inputs are identified. It is also one of the important phases of the system development activities.

- The system offers greater levels of user friendliness combined with greater processing speed. Therefore, the cost of maintenance can be reduced. Since, processing speed is very high and the work is reduced in the maintenance point of view management convince that the project is operationally feasible.

3.3 BEHAVIOURAL FEASIBILITY

People are inherently resistant to change and computer has been known to facilitate changes. An estimate should be made of how strong the user is likely to move towards the development of computerized system. These are various levels of users in order to ensure proper authentication and authorization and security of sensitive data of the organization.

- This software is of great use to persons using airline facilities frequently. This software will of course ease the airline booking process. And, Airline authorities will find it handy to set their airlines ticket fare and maximize their economic profitability. And, the back-end server is floated on Google – cloud platform. Thus, NO hassle of maintenance. The entire maintenance task is handled at scale by Google cloud platform. And, GoIbibo developer API is maintained by GoIbibo developer community.

Chapter 4

System Testing and Implementation

4 System Testing and Implementation

4.1 Test Case Design

4.1.1 Black Box Testing

Black-box testing (also known as functional testing) treats software under test as a black-box without knowing its internals. Tests are using software interfaces and trying to ensure that they work as expected. As long as functionality of interfaces remains unchanged, tests should pass even if internals are changed. Tester is aware of what the program should do but does not have the knowledge of how it does it. Black-box testing is most commonly used type of testing in traditional organizations that have testers as a separate department, especially when they are not proficient in coding and have difficulties to understand the code. It provides external perspective of the software under test.

Some of the advantages of black-box testing are:

1. Efficient for large segments of code
2. Code access is not required
3. Separation between user's and developer's perspectives

Some of the disadvantages of black-box testing are:

1. Limited coverage since only a fraction of test scenarios is performed
2. Inefficient testing due to tester's luck of knowledge about software internals
3. Blind coverage since tester has limited knowledge about the application

If tests are driving the development, they are often done in the form of acceptance criteria that is later used as definition of what should be developed. In that case black-box testing relies on some form of automation like Behavior Driven Development.

4.1.2 White Box Testing

White-box testing (also known as clear box testing, glass box testing, and transparent box testing, and structural testing) looks inside the software that is being tested and uses that knowledge as part of the testing process. If, for example, exception is thrown under certain conditions, test might want to reproduce those conditions. White-box testing requires internal knowledge of the system and programming skills. It provides internal perspective of the software under test.

Some of the advantages of white-box testing are:

1. Efficient in finding errors and problems
2. Required knowledge of internals of the software under test is beneficial for thorough testing
3. Allows finding hidden errors
4. Programmers introspection
5. Helps optimizing the code
6. Due to required internal knowledge of the software, maximum coverage is obtained

Some of the disadvantages of white-box testing are:

1. Might not find unimplemented or missing features
2. Requires high level knowledge of internals of the software under test
3. Requires code access

White-box testing is almost always automated and in most cases has the form of unit tests. If done before the development, it takes the form of Test Driven Development (TDD).

4.2 Test Execution

4.2.1 Unit Testing

While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.

Test Case	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Check whether model gives prediction	Match the result of running request.py and the predicted fare.	Yes	Both results should match	As Expected	Pass
2	Check whether GoIbibo API fetches results.	Clicking on Search button	Yes. List of flights	Table of ~50 flights status and their Fare.	As Expected	Pass
3	Check whether Graphs plotted on Dashboard	Check for subscription topic from cloud SDK Shell. Select node on Dashboard.	Yes	Graph for Humidity and temperature of each node. And, its Heat map over SVG.	As expected	Pass

4.2.2 Integration Testing

Even if the units of software are working fine individually, there is a need to find out if the units if integrated together would also work without errors. For example, argument passing and data updating etc.

Test Case	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result
1	Check whether Web page loads on URL search.	Type in URL in search bar and click search.	Page loader	Successfully loading	As Expected
2	Check each navigation button is responding	Click on each navigation button and floating scroll button.	Sub sections of web page loading efficiently. And, all modules working perfectly.	Works Perfectly.	As Expected

4.2.3 System Testing

The software is compiled as product and then it is tested as a whole. This can be accomplished using one or more of the following tests:

- **Functionality Testing:** Tests all functionalities of the software against the requirement.

- **Performance Testing:** This test proves how efficient the software is. It tests the effectiveness and average time taken by the software to do desired task. Performance testing is done by means of load testing and stress testing where the software is put under high user and data load under various environment conditions.

- **Security and Portability:** These tests are done when the software is meant to work on various platforms and accessed by number of persons.

Test Case	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Check input field responses in Pop –up window.	Validate input data and data on pop – up report.	Table entries about trip	Data matches.	As Expected	Pass
2	Correct flight status fetching	Validate input date, source, destination matches with the output record list.	Source, destination and date of journey same.	Data valid.	As Expected	Pass

3	Multiple nodes selection on dashboard.	Select more than one nodes at a time	Multiple area charts on dashboard.	Overlaying colors of area chart.	As Expected	Pass
4	Heat map on SVG	Select different nodes and time period.	Change in SVG gradient colors.	Chart gradient Varies.	As expected	Pass

4.2.4 Regression Testing

Whenever a software product is updated with new code, feature or functionality, it is tested thoroughly to detect if there is any negative impact of the added code. This is known as regression testing.

4.2.5 Acceptance Testing

When the software is ready to hand over to the customer it has to go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and if user does not like the way it appears or works, it may be rejected.

- **Alpha Testing:** The team of developer themselves perform alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs.

- **Beta Testing:** After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. This is not as yet the delivered product. Developers expect that users at this stage will bring minute problems were skipped to attend.

Chapter 5

Conclusion

Result

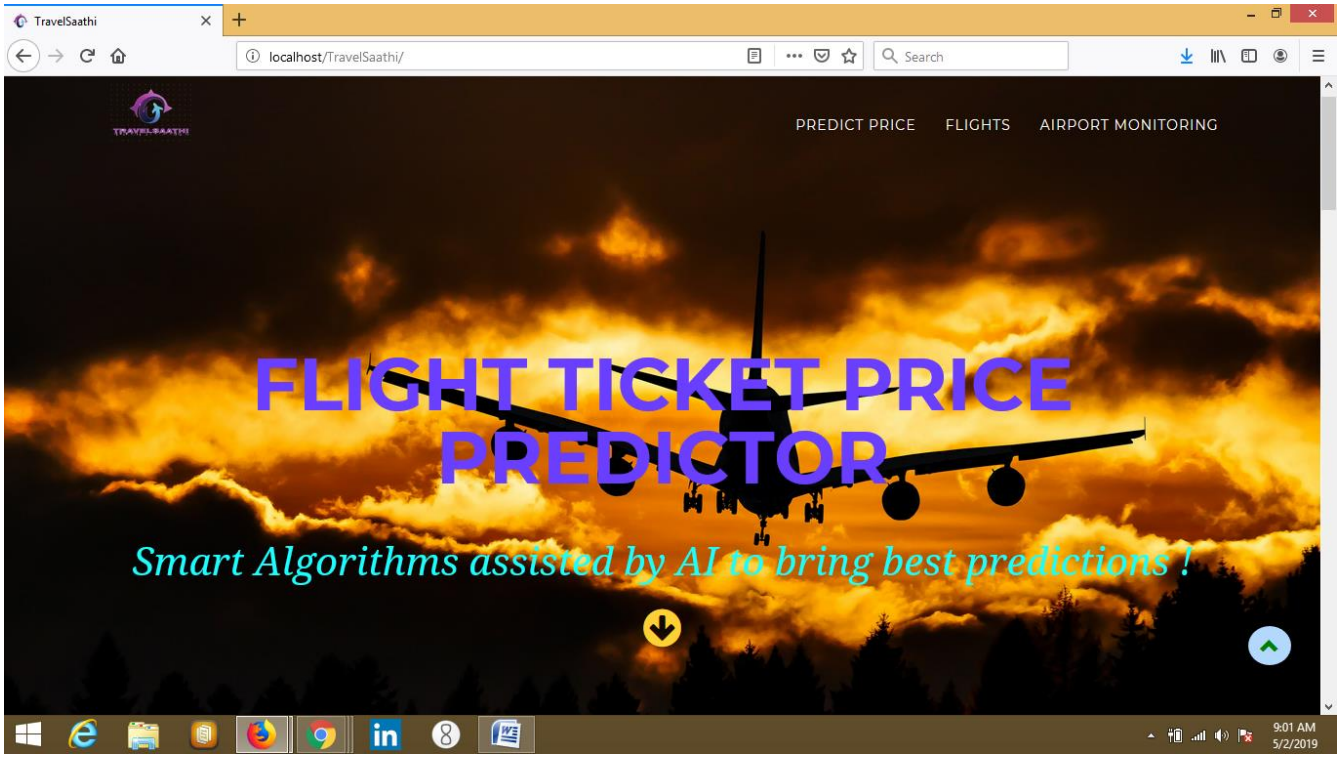
TravelSaathi is an intelligent AI assisted web platform to help aviation individuals take smart, critical and environmental friendly decisions. It is mainly developed to help aviation companies and airport authorities. We can intelligently predict flight ticket price, and can compare it with competitive flight services in the market. The Machine Learning model is working in the back-end to support prediction. It is highly optimized ensemble regression model. One can also visualize the Heat index and monitor all over the airport premises. This could be used to provide sustainable Packages and Food Storage. Also, it can be used to provide environmental friendly solution to electronic appliances. Viz. setting mode and temperature of AC.

Thus, this multi- purpose application is versatile in usage. It is also built using multiple cross compiled languages. And, data is stored on cloud to avoid any loss. This web app can be very handy in aviation industries.

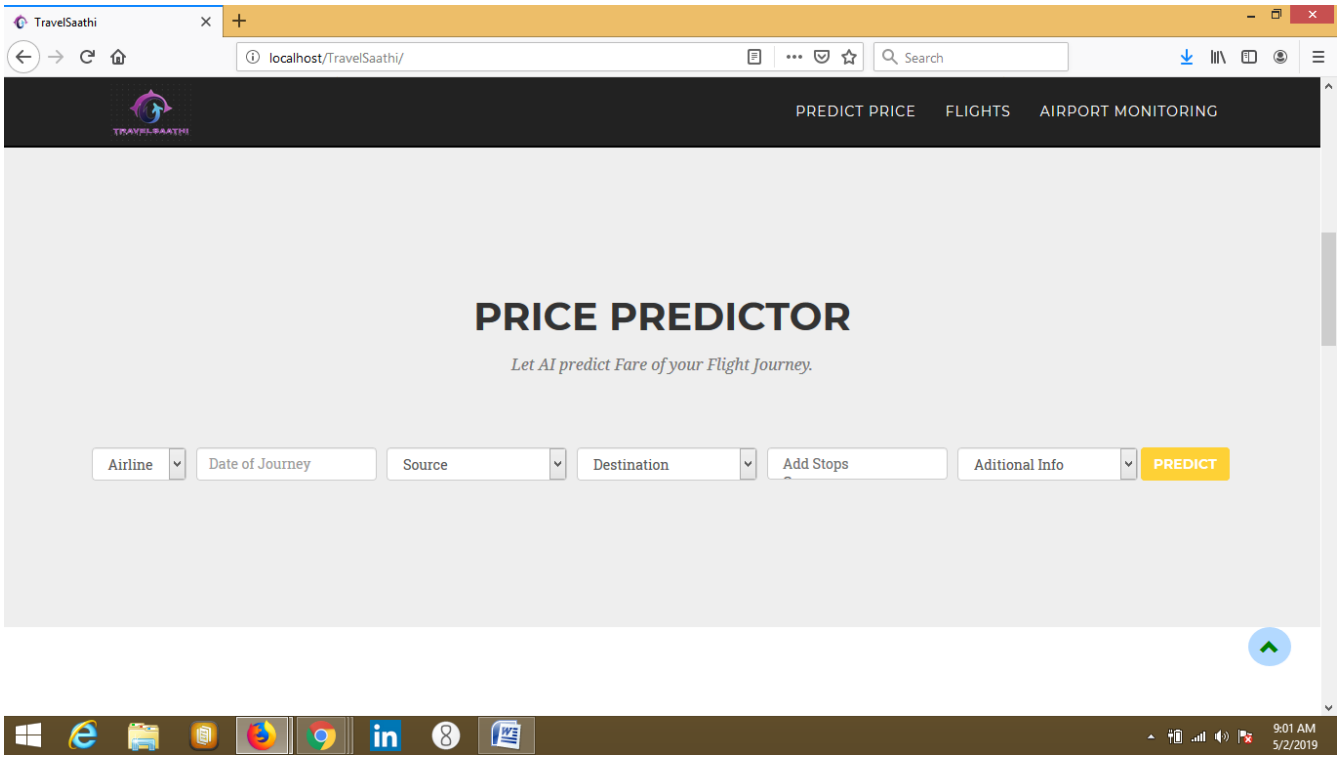
Future Scope

This can be further enhanced to bring the optimal Fare price to the other side of end user i.e. the customer. Even customers can compare prices of similar trips and can chose best option for them. The heat index feature could be used to provide IIoT 4.0 (industrial IoT) to airport premises. And, temperature and modes of Air conditioners can be controlled automatically using Artificial Intelligence. Customers can avail best flight option at minimal cost. Baggage and packages could be tracked throughout the intra - airport premises and inter – airport premises.

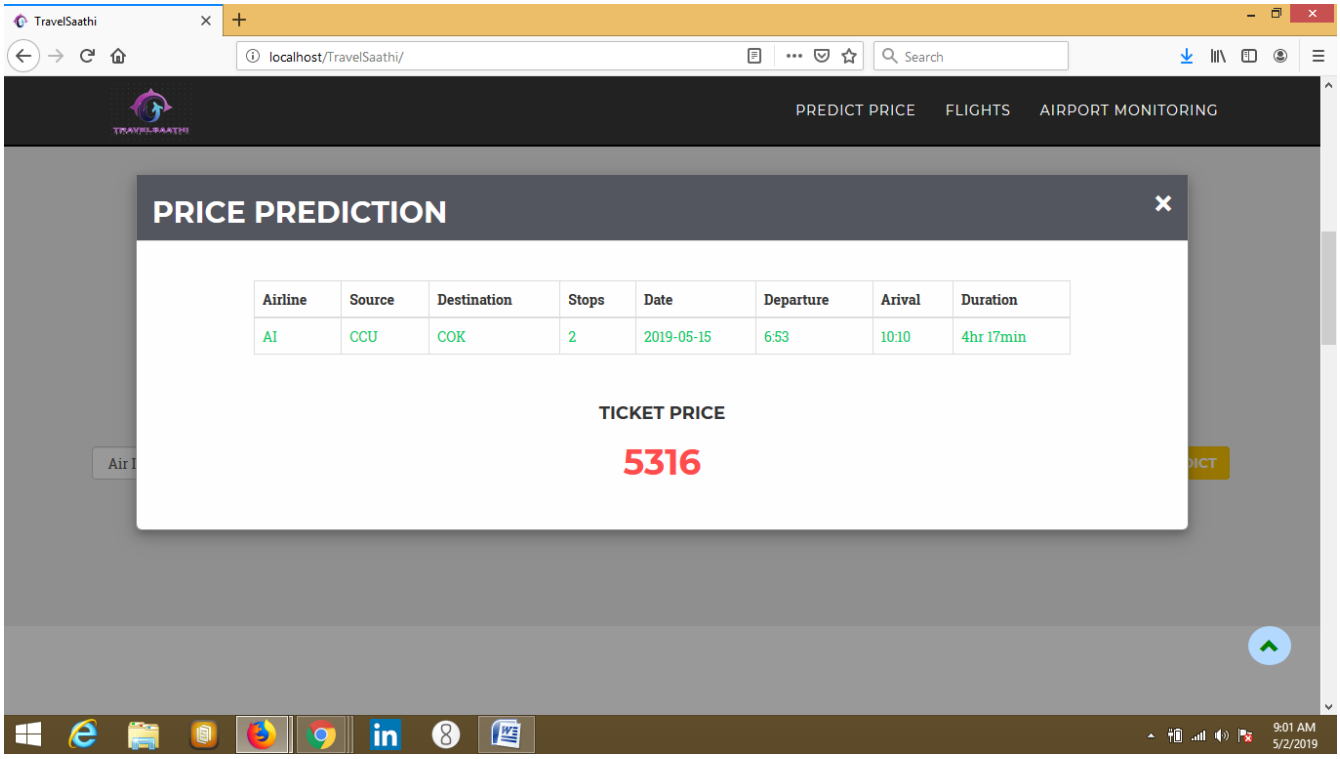
Screenshots & Result



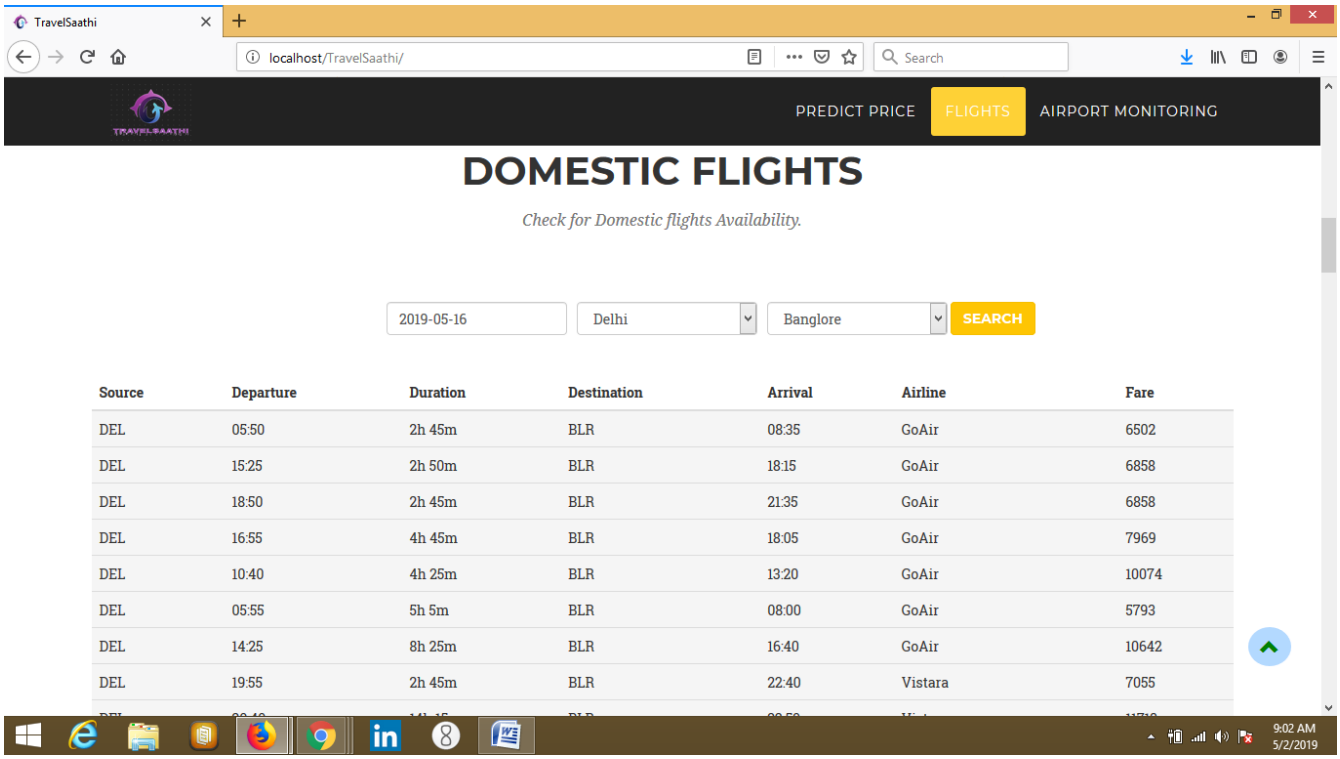
Welcome screen of TravelSaathi



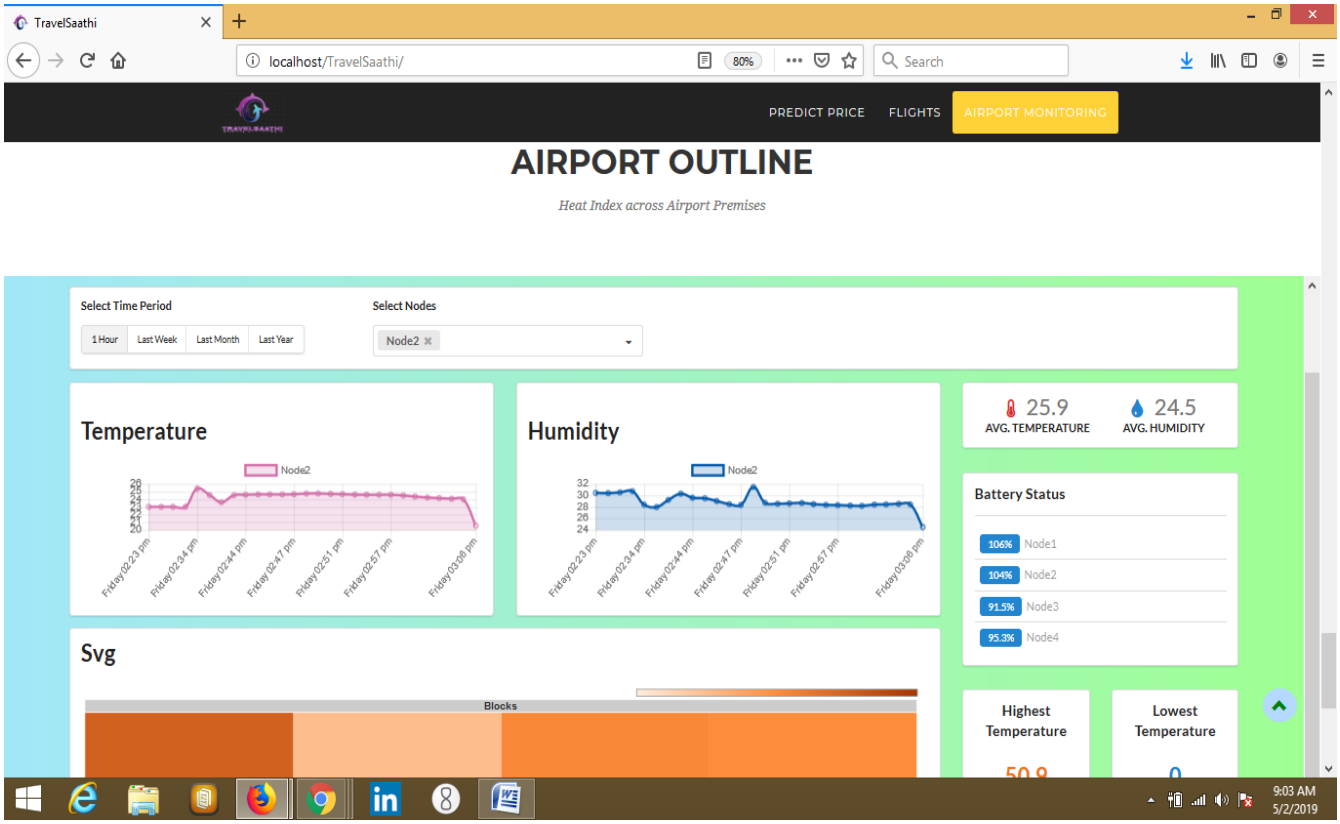
Input fields to predict Flight ticket Price



Predicted Ticket Price



Check Competitive Flight Status



Heat Index of Airport Premises

7. Bibliography

- https://www.w3schools.com/js/
- https://cloud.google.com/iot/docs/
- https://cloud.google.com/pubsub/docs/
- https://firebase.google.com/docs/functions/
- https://www.w3schools.com/php/
- https://cloud.google.com/bigquery/docs/
- https://www.kdnuggets.com/2019/01/build-api-machine-learning-model-using-flask.html
- https://www.flaskapi.org/
- https://developer.goibibo.com/docs