Trip Estimation

A MINI PROJECT REPORT

submitted by

Krishneth A (715520104025)

Shrishti Krishna (715521104042)

Kamal S (715521104020)

Kamal Chander (715521104021)

Department Computer Science and Engineering

PSG Institute of Technology and Applied Research Coimbatore

in collaboration with

SAP Labs, Bengaluru

MAY 2023

BONAFIDE CERTIFICATE

Certified that the mini-project titled **"Trip Estimation"** is the bonafide work of **"KRISHNETH A (715520104025), SHRISHTI KRISHNA S (715521104042), KAMAL S (715521104020), KAMAL CHANDER (715521104021)"** which is carried out under our mentoring.

SIGNATURE

INDUSTRY MENTOR
ANSHIKA MISHRA
Spend Engineering,
SAP Labs
Bengaluru,
Karnataka, India

SIGNATURE

ACADAMIC MENTOR
Dr. S. Lokesh
Associate Professor
Computer Science and Engineering
PSG Institute of Technology and Applied Research,
Coimbatore – 641 062

SIGNATURE
Srivatsan Santhanam
VICE PRESIDENT
Spend Engineering,
Head of Concur R&D,
SAP Labs
Bengaluru,
Karnataka, India

SIGNATURE
Dr. R. Manimegalai
HEAD OF THE DEPARTMENT
Professor
Computer Science and Engineering
PSG Institute of Technology and Applied Research,
Coimbatore – 641 062

Submitted for evaluation on _____

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to our beloved Principal, **Dr. G. CHANDRAMOHAN, B.E. (Hons), M.Tech, Ph.D,** for his overwhelming support and encouragement on this project.

We extent our indebted to **Mr. SRIVATSAN SANTHANAM,** Vice President, Spend Engineering, Head of Concur R&D, SAP Labs India for the opportunity and support which was instrumental in the completion of this project.

We are greatly indebted to **Dr. R. MANIMEGALAI, M.E, Ph.D** Head of the Department, Computer Science and Engineering for her guidance and continuous support which was instrumental in the completion of this project.

We extend our thanks to our industry mentor, **Ms. ANSHIKA MISHRA,** SAP Labs, Bengaluru, India and academic mentor **Dr. S. LOKESH, M.E, Ph.D** Associate Professor, Computer Science and Engineering for his technical support and constant supervision without which we could not have completed this project study.

KRISHNETH A

SHRISHTI KRISHNA

KAMAL S

KAMAL CHANDER

ABSTRACT

This project aims to develop a website which makes it easier for anyone to find out about the cost of a business trip to any city inside India. We aim to do this using a machine learning model which can predict the cost of trip when it is provided with details regarding the trip such as the destination, number of days, mode of transport etc. The application will consist of many algorithms each of which will predict the cost of a singular part of the trip like the hotel or the transport.

Each of these models will be trained on a large dataset which will be cleaned and preprocessed. Then all the specific features will be extracted and used to train the models. As we want to predict a numerical value using a model with values we feed, a supervised algorithm such as logistic regression or K-Nearest Neighbor regressor is ideal for this project as we want the model to learn from the data and predict the approximate value.

The application will be used by employees to submit application with their estimate, a approver can go over the application and using the value generated by the model as reference he/she can make a decision whether to approve the application or not.

All of the above functionality will be implemented to the fronted of the website, which will be built using a JavaScript library like react.js. The fronted will provide an aesthetically pleasing user interface for the user. Employees and approvers can register and login in to the website and will be directed to their respective pages.

Keywords: Machine Learning, K-Nearest Neighbor, React.js, Supervised Algorithm

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface.
CSS	Cascading Style Sheets
DOM	Document Object Model
IDE	Integrated Development Environment
JS	JavaScript
KNN	K Nearest Neighbors
ML	Machine Learning
ND	N Dimensional
SKL	SciKit Learn
UI	User Interface

CHAPTER 1

INTRODUCTION

1.1 PROJECT MOTIVATION

Travelling expenses can always be more than what we expect, and often we might face issues which we could have avoided if we had known the cost of the trip ahead of time, this is especially the case for business trips.

With the presence of an estimate of how much a trip will cost, People who are travelling can make informed decisions on all aspects of the trip. For example it will allow a person to decide if he/she wants to travel by airplane or by train or if they want to stay at a 3 or 4 star hotel. The estimate can even include necessary expenses which one might forget to account for, such as parking, wifi or nearest airport and domestic travel costs within the city.

This application also allows superiors to monitor and approve the budget for a trip for their employees. If the estimate provided by the estimate is too expensive then they can be asked to revise their budget. This allows a company to implement cost control measures as the estimate provided by the employees can be compared to make sure it is valid and not exceeding the allowed budget by too much.

1.2 PROBLEM STATEMENT AND OBJECTIVES

1.2.1 PROBLEM STATEMENT:

The primary objective of any company is to control company expenses. Trip estimates are particularly useful to approve, reject or modify a trip. This also helps in budgeting for the trip. The current process allows users to enter the requested amount for a trip, but there is no way to compare it to a trip estimate. Adding a trip estimator feature can help in notifying the approvers about the trip estimated amount and the requested amount difference, which can prompt them to ask the user to reassess the amount requested based on the trip estimator.

1.2.2 OBJECTIVES:

- A website which accepts the necessary details of the trip such as number of days, transportation mode, source and destination, hotel where one will be staying.
- User Interface is built using react.js
- Backend algorithms built using python.

1.3 SCOPE AND LIMITATIONS OF THE PROJECT

1.3.1 SCOPE:

1. User Interface: An aesthetically pleasing yet simplistic interface for the user to login in and to make applications for a trip by providing details such as destination and mode of transport and for the approver to accept or reject an application.
2. Data Integration and cleaning: The algorithm to be created is to be detailed and catered for the specific purpose of estimating the cost. Necessary changes are to be made so that the data is more refined for training a model with.
3. Cost Estimation: Machine learning algorithms which look at the data provided and give an accurate estimate of the cost of the total trip. The application will consist of many models which will each provide an estimate for a segment of the trip.
4. Multiple Transport Options: Provide users with a range of transportation options, such as flights, trains, and rental cars, along with their respective costs. the most suitable one from the given options is to be selected.

1.3.2 LIMITATIONS:

1. A trip consists of too many variable aspects that it is not possible for us to get an extremely accurate estimate of the trip.
2. The accuracy of the algorithms heavily depends on the data which is used to train the model, if the details of the trip do not coincide to the one in the dataset then the estimate generated will not be accurate.
3. There are always unforeseen events which take place in a trip which can dramatically increase the cost of the trip.
4. The algorithm does not take into account the personal preferences of each individual in the trip, and the estimate is generated with economy in mind. This may suggest a estimate which excludes certain features of the trip, such as a train with AC or a direct flight instead of an indirect one which is in most cases cheaper.

CHAPTER 2

PROJECT ARCHITECTURE, DESIGN AND IMPLEMENTATION

2.1 SYSTEM ARCHITECTURE

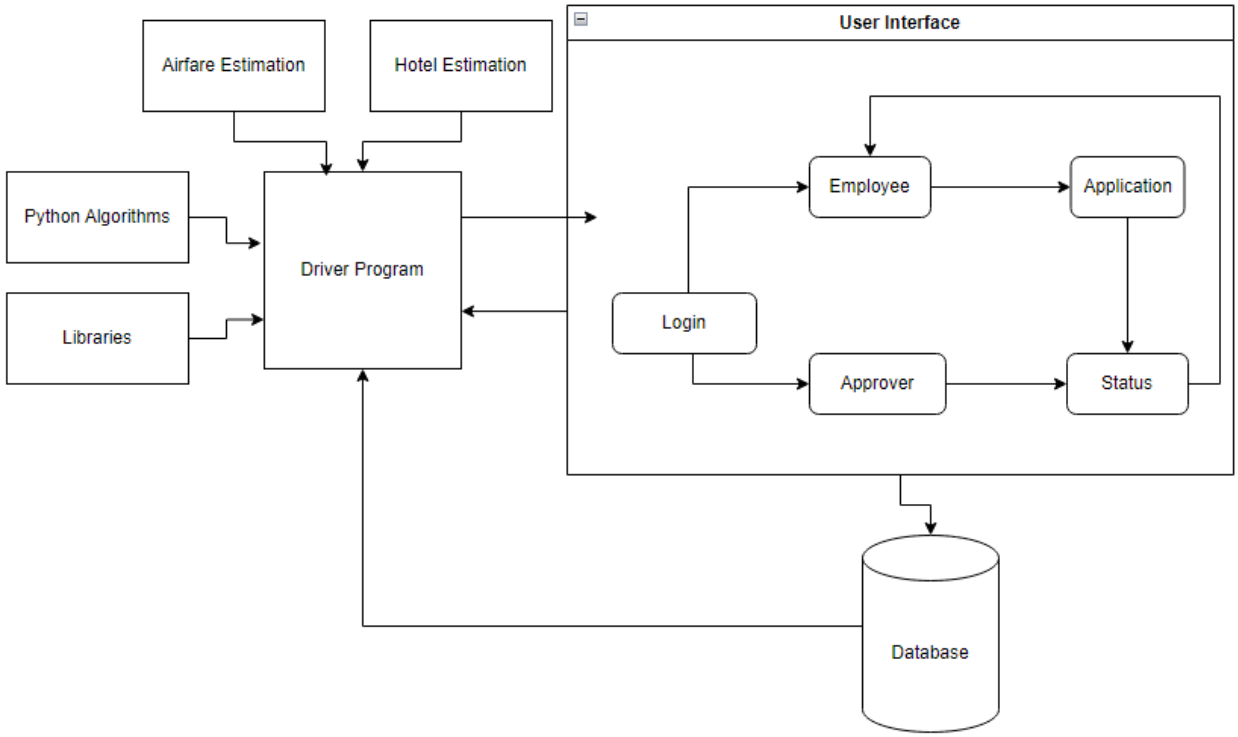

Fig 2.1 System Architecture

The above figure represents the overall system architecture, it consists of a user Interface or the fronted which acts as the interactive interface using which the employee or approver can perform their necessary tasks such as requesting or approving respectively. The user interface is connected to the backend which consists of multiple algorithms each of which can predict the cost of a part of the trip. The algorithms are connected in the driver program where all the necessary libraries are imported.

The fronted and the backend are connected to a database where data is stored and retrieved whenever necessary.

2.2 OVERVIEW OF THE DESIGN PROCESS

Flow Chart:

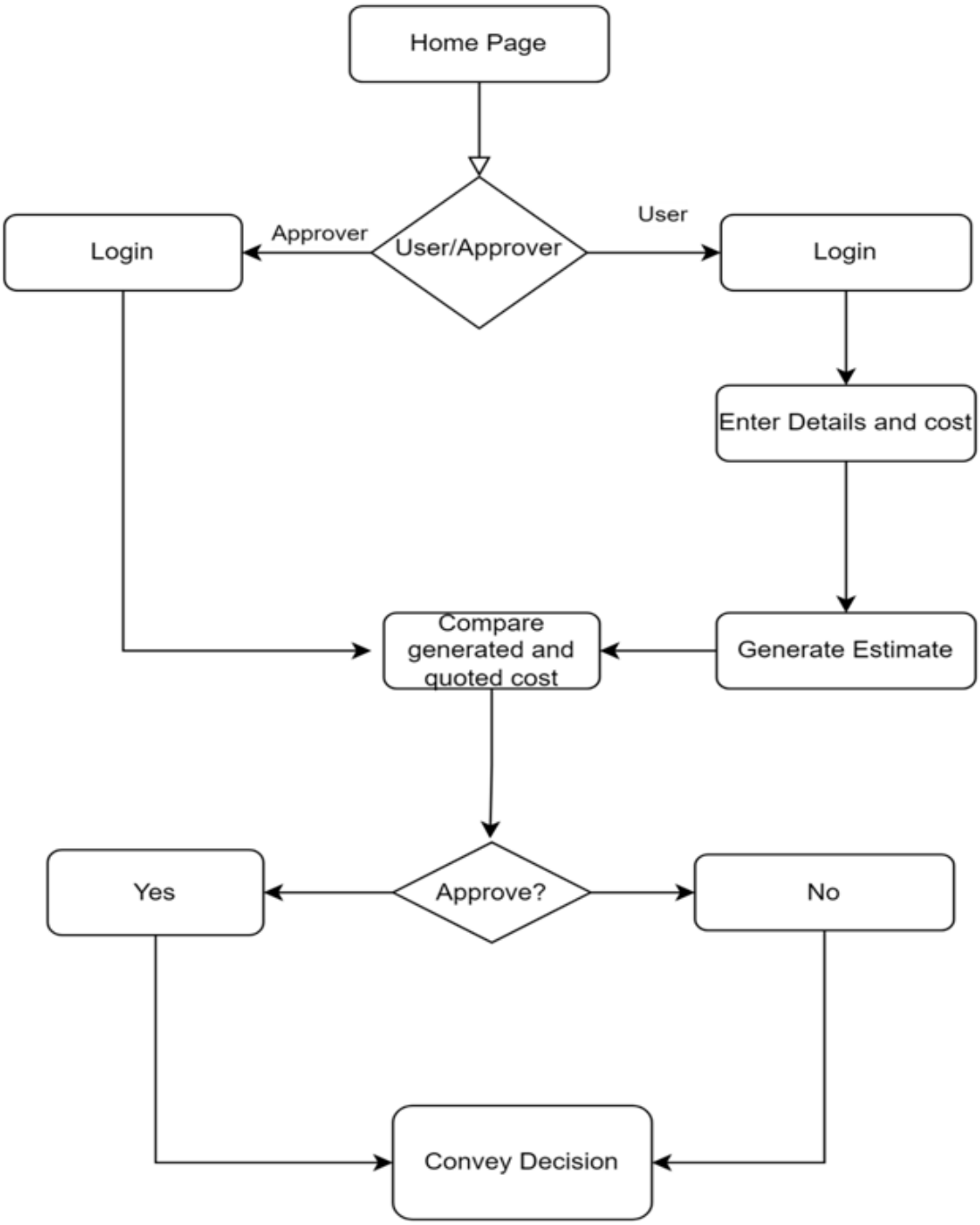

Fig 2.2 Flowchart Of Overall Application

The flow chart above, shows the step by step flow of control of the application, initially the employees makes a request for a trip by providing the details and the estimate. Then an estimate is generated by the algorithm, both of these are received by the approver who makes a decision whether to approve or not. If the request has been rejected then a message is passed to the employee who made to request to make a revision to the budget.

2.2.1 User Interface:

This is the interactive interface the user sees when he accesses the website. The user interface allows a user to do any of the following:

- Register: If the user is new to the website then he can create a account by registering, this stores the username and the password of the user in the database.
- Login: If the user is someone who has previously registered, then he can simply login by providing his username and password, upon which verification is performed. Once verified the user is directed to his page. If the user is an employee, then he can provide details of the trip and apply for approval from the approver whilst providing an estimate of his own. This application is sent to the approver. If the user logging in is an approver then he is redirected to the pending requests page where he can see the entire request list which is pending. The approver can see the estimate generated by the algorithms and if the estimate received from the employee is valid and acceptable then the approver can accept the request or if it is exceeding the limit by a lot then he can reject it with a message asking for revision of the amount.
- Status: After the decision has been made by the approver, the employee is made aware of this decision through the status page where he can see the status of his request.
- The interface should also display the estimated travel time and distance once the user submits the query.

2.2.2 Data Source:

- The estimation of the cost of the trip is done by the algorithms running in the backend. These algorithms rely on the datasets provided to them to make a prediction.
- The machine learning algorithm which predicts hotel prices relies on a dataset consisting of the hotels with their features and their prices. The same goes for the model which predicts airfare costs. These algorithms primarily rely on historical data which they analyze to give a prediction with acceptable accuracy.
- The algorithms which predict the cost of travel via car or train depend on data which provide information on the toll costs, and the fuel costs.

2.2.3 Database:

All the data regarding the user's login details are stored here upon registration and are retrieved when necessary. The list of request for the approver and the status of the request for the employee are also stored in the database. It is connected to the user interface for the purpose and to the backend driver program for the purpose of calculation of the amount by the machine learning algorithms.

2.2.4 Backend (Algorithms):

- A backend system will handle the processing and calculation tasks.
- It will manage communication with external APIs or services to fetch real-time data and integrate it into the estimation process.
- The system will store relevant data, such as user preferences, historical information, and cached results, to optimize future estimations.
- All the details collected at the user interface are sent to the backend of the application, here multiple algorithms access this data and perform the necessary predictions and provide a result.

- The different libraries required to run the driver program are imported and the necessary functions are also imported from the packages.
- The algorithms include a machine learning model for predicting the hotel price, which accepts the star rating, special features and location to predict the price. For transportation, a machine learning model is used for the airfare and when it comes to car or train a non machine learning algorithm is designed to perform the prediction. The transportation algorithms take the mode of transport as input along with source and destination to predict a value.
- When all the different models have predicted values they are summed into one and sent to the fronted to be displayed to the user. The user can refer to this value to either proceed with the application or revise their model.

2.2.5 Integration:

After the fronted and the backend have been created separately they are then integrated together along with the database.

Apart from the integration of the application itself, it can be integrated with other applications such as mobile apps to make the app on the go.

The system is built with scalability in mind; it can handle many users in a company and also can be extended to individuals outside of a company as a standalone application.

2.3 EXPLANATION OF THE ENGINEERING PRINCIPLES USED IN THE DESIGN

Python:

The entire backend of this project is run on python. Python is an open source high level language with its main benefit being that it allows the importing of a vast number of libraries to assist the programmer in developing an application. These libraries help out the programmer immensely as they reduce the need to code an exorbitant amount of boiler plate code which is a waste of time. They also allow for us reuse pre-built code which suit our application.

Python best suits this application as many machine learning algorithms and libraries are built on python. Some examples are the jupyter notebook IDE which is a development environment which uses python to help programmers built complex machine learning models. Python also makes it very easy to handle large datasets inside of the IDE itself instead of having to pre-process the data externally and then import it into the application.

Machine Learning:

There are various libraries which we are required to import in order to build the machine learning model we want. Some of these libraries are:

NumPy:

It is an open source python package which allows us to handle multi-dimensional arrays called nd arrays, which stands for n dimensional arrays. These arrays are used to process the data and train it.

Pandas:

It is a package which is a works along with the NumPy package to provide the programmer with various data manipulation and visualization techniques.

With the help of Pandas and NumPy we can perform the following operations:

- Import a data set as a csv file into the integrated development environment.
- Convert the data into a format known as DataFrame which allows us to manipulate the data.
- Perform necessary operations in order to clean the data such as, remove duplicates, drop unnecessary columns and rows. Add new columns which provide further information to the machine learning model. These new columns contain data which are the data from previously present columns that have been manipulated to better suit our need.
- Employ various data visualization techniques such as graphs, matrices, pie charts and etc. These methods enable us to look at the various patterns in the data which we are required to understand to build a model with acceptable accuracy.

Matplotlib:

It is a library that provides a multitude of methods for us to represent data as plotted graphs, for better understanding of the pre and post trained data. It provides comparisons graphs for understanding the performance of the model with the test set. It also provides comparison graphs for accuracy of different models trained on the same dataset.

Sklearn:

It a python package which provides various data analysis methods and function for us to train our data. Using this package we are able to choose from a plethora of pre-built machine learning algorithms the one which best suits our purpose. These per-built algorithms are imported and the data we require to be analyzed is fitted to these models and the custom model is trained.

Functions imported from sklearn:

- train_test_split- Dividing the dataset into train and test set according to given ratio.
- confusion_matrix – Break down of how well the data has been categorized

- accuracy_score- Gives information on how accurate the trained model when tested with the test set. It is given as a percentage.
- LinearRegression- Machine Learning algorithm to train models for linear dependant data.
- mean_squared_error- error loss function.
- KNeighborsRegressor - Machine learning algorithm for non linear data.
- StandardScaler- used in scaling the data appropriately.

KNeighborsRegressor:

The KNeighborRegressor is a regression algorithms, regression is a supervised machine learning algorithm where the data is fed by the programmer and the model is expected to learn to identify the pattern or the different relationships in the data and replicate this on new data and provide results. For example, we want to understand the relationship between the star rating of the hotels and the price, we train the model with input as star rated hotels and their respective price and in essence teach it to predict the price when the star rating is given.

KNeighborsRegressor can be used to predict continuous data i.e. data which does not have fixed set of values, instead it can be anything in a given range. In this case the price of each hotel falls into a fixed range, and not a set of pre-fixed values.

The working of the algorithms can be understood by taking a graph into consideration on which each individual value form the data set is plotted and given a weight. The training process consists of plotting all the data as points in the graph, where these points go depend on their features and the weight assigned to them. Once the training data has been plotted the model has been built. When we want the model to predict a new data we simply plot it as a new point. Then the distance between the new point and the closest few point are measured and their average is calculated. This average is combined with the weights of the existing points to provide the result of the new point. The parameter K signifies how many of the closest points are to be taken into consideration when calculating the average. If the graph is quite compressed i.e. all the points lie

close to each other, even if in different groups, then the value of k can be increased to provide better accuracy. But is the points are all spread out then the value of K must be restricted to a lower value. In most cases the value of K is anywhere in the range 5 -10.

Here is the working of the KNN algorithm in a step wise manner:

Step 1: Decide on the neighbours' K-numbers and calculate the Euclidean distance between K neighbours

Step 2: Based on the determined Euclidean distance, select the K closest neighbours.

Step 3: Count the number of data points in each category among these k neighbours.

Step 4: Assign the fresh data points to the category where the neighbour count is highest.

React.js:

React.js is a JavaScript library primarily used for building webpages on the internet . React allows us to create reusable UI components. It is a declarative library i.e. when we want to create a webpage we simply describe how the UI is supposed to look and React will make it happen on its own, this makes the tedious and repetitive part of implementing a design very simple. Whenever we want to make changes of update to our webpage, we can do it in an efficient manner using React where only the component which needs to be update is modified and the other parts are left untouched.

React has reusable components which are small, self-contained pieces of code that can be reused throughout an application. Thanks to these reusable codes it is very easy to maintain and modify the webpage without affecting the entire page.

DOM in React: In React there exists an virtual copy of the actual DOM or the Document Object Model, this virtual DOM acts as an interface to the actual DOM when changes are made to the pages, the virtual DOM is updated first, and only then is the actual DOM updated, this allows for a buffer that can reduce the processing required as only the part which need updating is done so in the actual DOM this is realized from the virtual DOM.

Hook Function: React offers a unique function called the hook which makes the addition of react features to functional components.

State and Props: React.js manages the state of a component using the state object. The state represents the internal data of a component that can change over time. Props (short for properties) are used to pass data from a parent component to its child components. Props are read-only and cannot be modified by the child components.

Lifecycle Methods: React.js provides a set of lifecycle methods that allow developers to hook into different stages of a component's life cycle. These methods include componentDidMount, componentDidUpdate, and componentWillUnmount, among others. They enable developers to perform actions like initializing state, fetching data, or cleaning up resources.

In react the information always flows from the parent to the child component and not the other way down. This is known as unidirectional flow.

2.4 DESCRIPTION OF THE STEPS TAKEN TO IMPLEMENT THE PROJECT DESIGN

BUILDING THE ALGORITHM FOR HOTEL PRICE ESTIMATION AND AIRFARE ESTIMSTION.

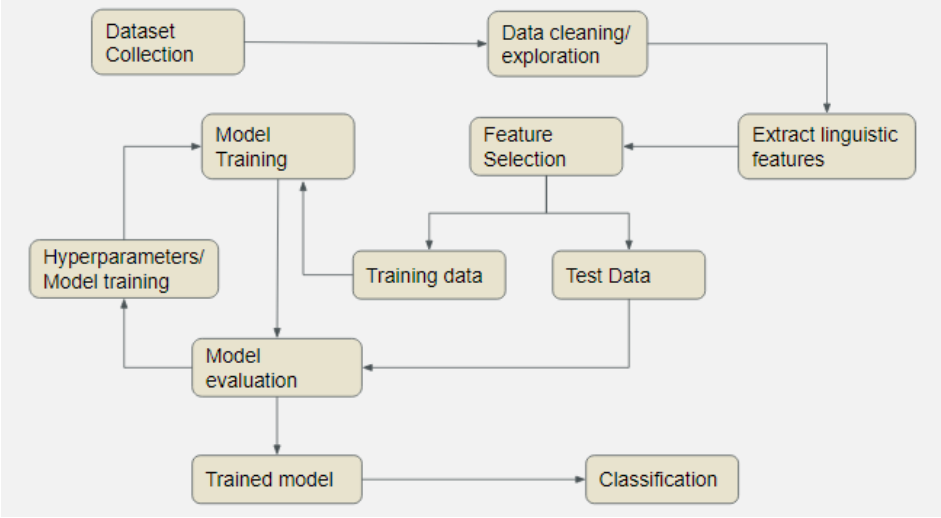

Fig 2.3 Architecture of ML Model

1. Preprocess the data by removing unnecessary columns which do not contribute to the price of the hotel, example name, etc.
2. Split the data into training, validation, and test sets.
3. Create a vocabulary of location, star rating and facilities of the hotels from the training set and assign each of those an index depending on how they affect the price.
4. Convert the headlines and labels into sequences of indices using the vocabulary.
5. Pad or truncate the sequences to a fixed length.
6. Model Initialization: Import the required libraries, including scikit-learn, and initialize the KNeighborRegressor model. Specify the hyperparameters such as the number of neighbors (K) and any other relevant parameters.
7. Train the model i.e. fit the KNN algorithm to the dataset and makes the plotting graph.
8. After the model is done training validate its accuracy by running the test dataset on the model and checking for the accuracy.
9. After validation is performed if the results are not satisfactory then rebuild the model with necessary changes.

8. Tune the hyperparameters in order to achieve the optimal result.

Test the model by feeding the test sequences and labels and computing the test loss and accuracy. Report the results and compare them with other models or baselines.

SETTING UP ENVIRONMENT

Installing pip, a Python package manager, is the first step. Every Python packagethat is listed in the Python Package Index can be easily installed using pip bytyping <pip install packagename>. The web application was created using the Flaskweb framework. Downloaded and set up is the PyCharm Integrated Development.

Environment (IDE) Jupyter Notebook. Activate the ML libraries.

- Numpy: for any work with matrices, especially math operations
- Pandas: Data handling, Manipulation, and Analysis
- Matplotlib: Data visualisation
- Scikit learn: Machine learning

```
import datetime
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

Fig 2.4 Hotel ML Model

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.neighbors import KNeighborsRegressor
from sklearn.preprocessing import StandardScaler
```

Fig 2.5 Airfare ML Model

Then the dataset is a collection of data which include different details of the trip such as the prices of different hotels in different cities and also the transport costs for different modes of transport, number of days and source and destination of the travel.

After the collection of dataset, we have to clean the data to make it suitable for training. The process of cleaning data involves manipulating the data to make it easier for us to work with it. We will first remove all the duplicates which will reduce the training time of the model and make it more efficient.

```
df.drop_duplicates(keep=False, inplace=False)
```

Fig 2.6 Removing Duplicate Data

The above function “drop” of the Pandas library allows us to do this with ease.

Check the dependencies of each of the features of the, in order to understand the relationships between each of them, this can be done by a matrix with the dimensions as the number of features

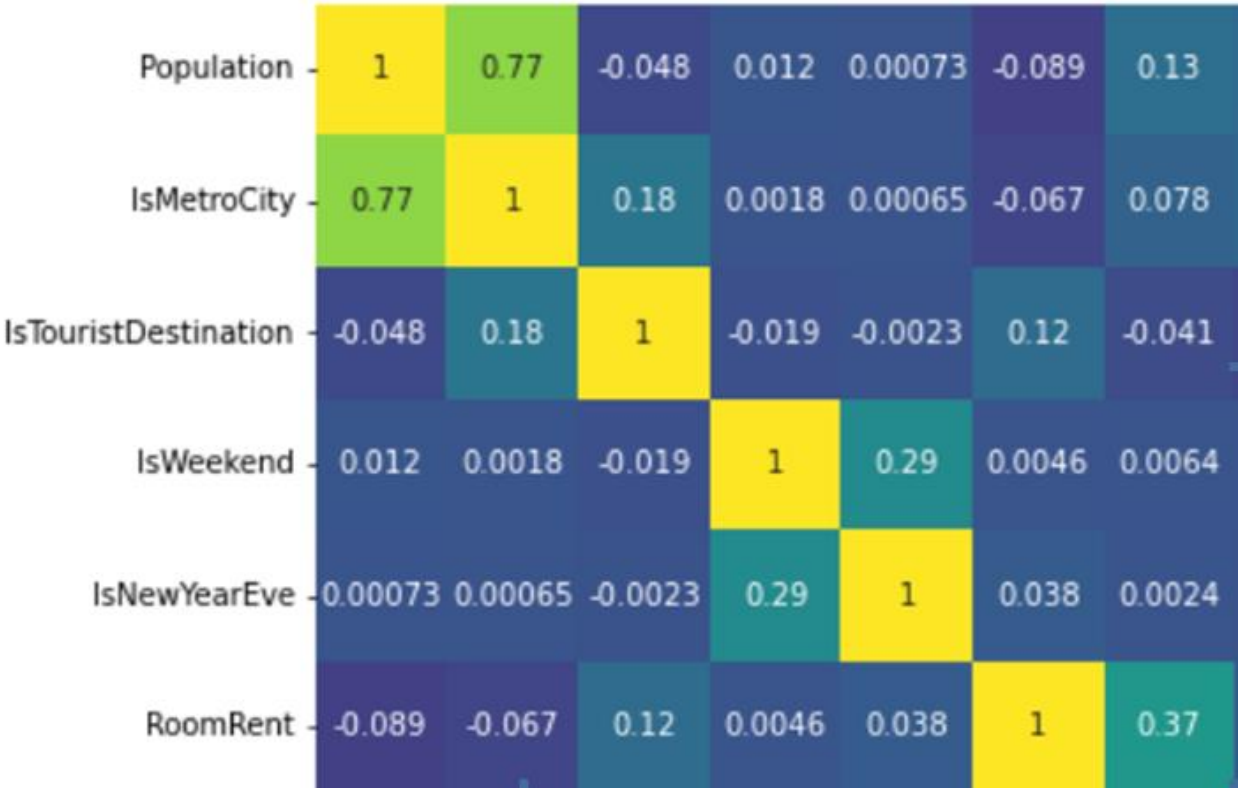

Fig 2.7 Matrix to represent relationships between features

The higher the percentage is the more dependent the features are on each other, this gives us a good idea of which features should be prioritized and which are not necessary and will only stand to increase the training time of the model.

Finally we can train the model with the pre-processed data. The machine learning algorithm used to analyze the hotels and the airplane fare is the KNN or the K Nearest Neighbors Regressor.

We give the value of the parameter K as 5 specifying that the value of 5 neighbours is to be taken when calculating the average.

The exact above procedure is followed to train the model to predict airplane fare.

Implementation of the algorithms for train and car price prediction for transport:

Car:

For the prediction of price for car travel I first obtained the distance between the cities that are to be included. Then now distance had been obtained, we had to find the cost of the fuel required

for the trip. Only the cost of petrol is taken into consideration. After averaging out the petrol price in the different cities this value was taken to be cost of fuel. Next we considered the average mileage(kmpl) of a car in India to 25. Then divide total distance by mileage to get total fuel required in liter. Multiply by cost of fuel per liter to find the total fuel cost.

For the toll cost Google maps was used. Google maps give the cost of toll when travelling by car between cities. Adding both the fuel and toll cost gives the total transport cost using car.

Train

For the train price prediction, we used the price for reserving tickets on trains given in IRCTC. As the cost of different trains travelling between the same two junctions is usually the same,. I the price for booking was considered to be third tier AC.

Developing the frontend of the application:

Before coding of the fronted using react.js, the basic design of the website must be prepared as it will act as the template for us to build the user interface. This can be accomplished using Figma, which is a design application used to style webpages. Figma doesn't offer any implementation or functionality; it just acts as the guide and allows us to proceed knowing what is next in line for the development of the user interface.

Steps in building the user interface using react:

Setting up the Development Environment: Install Node.js: React requires Node.js to be installed on the machine.

Understand React Components: React follows a component-based architecture. We can start by understanding the basics of React components, including functional components and class components. Components are the building blocks of the website's UI.

Creating additional components: Breaking down the website's UI into smaller, reusable components. Create new component files in the SRC folder, such as Header.js, Footer.js, etc. Importing these components into the App.js file and use them to build your website's structure.

We can use CSS or a CSS-in-JS solution like Styled Components to style components. We can create separate CSS files or use the inline style attribute within your components to define styles.

Adding Functionality with Props and State:

Using props to pass data from parent components to child components and defining props in parent components and pass them down to child components. This allows you to customize the behavior and appearance of components based on the data passed through props.

Use state to manage component-specific data that can change over time. You can define and update the state within class components using the setState method.

Testing and running the website:

Once we have finished implementing all the webpages we have to test if these webpages will run bug free in all possible scenarios. To ensure we use alpha testing where third party users use the website and try using all the functionality of the site. Any bugs discovered are rectified.

Deploying the Website:

After testing is done and we are satisfied with the performance we can deploy the website on the internet.

CHAPTER 3

RESULTS AND ANALYSIS

3.1 VALIDATION PROCEDURES

To ensure smooth running of the website validation procedures are performed:

Input validation: We test the website with various inputs which test the entire scope of the website. The boundary cases are especially tested to see the kinks in the website. For example we test the reaction of the website if invalid inputs or inputs which are technically correct but don't make sense are provided. The obscurities are noted if any are present.

Performance: The performance of the website is validated when it is overloaded and what happens if the session is abruptly interrupted. The system is be consistent and will terminate current transaction if there is any interruption.

Authentication: Since the website requires the user to login, the security is tested and made sure unauthorized access is not possible, at least with minimal effort.

3.2 TEST RESULTS

Accuracy of the machine learning models:

```
X_train, X_test, y_train, y_test = train_test_split(df2,df["RoomRent"], test_size=0.33, random_state=10)
knn = KNeighborsRegressor(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(r2)
```

```
0.7438840840000683
```

Fig.3.1. Accuracy of Hotel ML Model

```
X_train, X_test, y_train, y_test = train_test_split(df3,df["Price"], test_size=0.33, random_state=42)
knn = KNeighborsRegressor(n_neighbors=5)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
#print('Mean squared error:', mse)
print(r2)
```

```
0.7630732529398102
```

Fig.3.2. Accuracy of Airfare ML Model

User Interface:

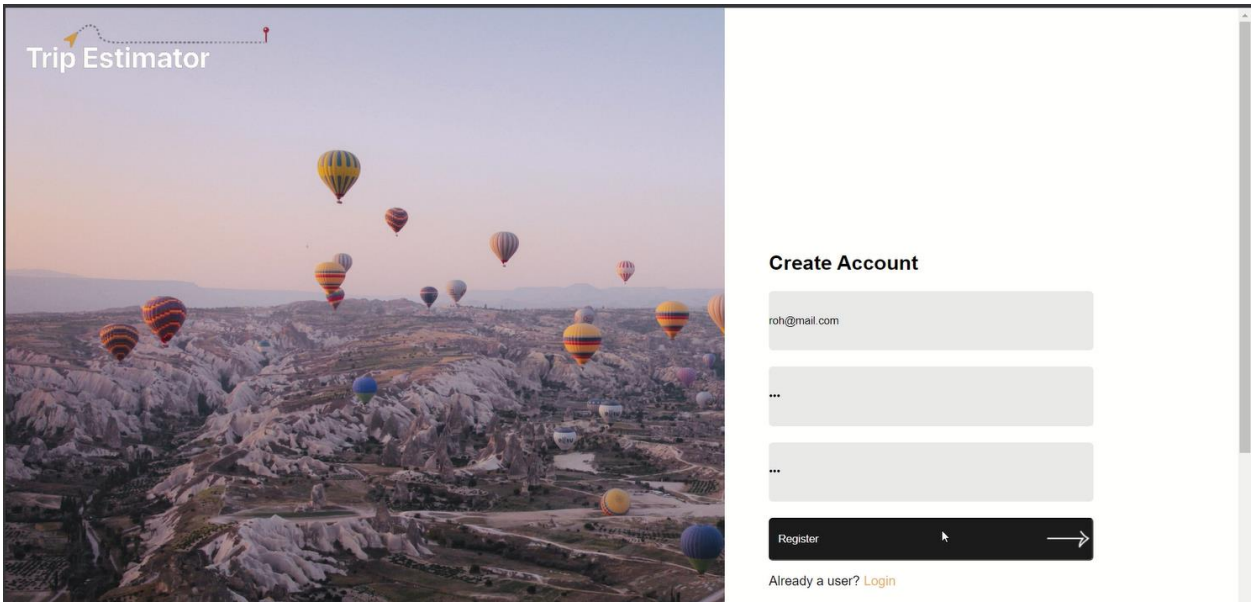

Fig.3.3.Register Page

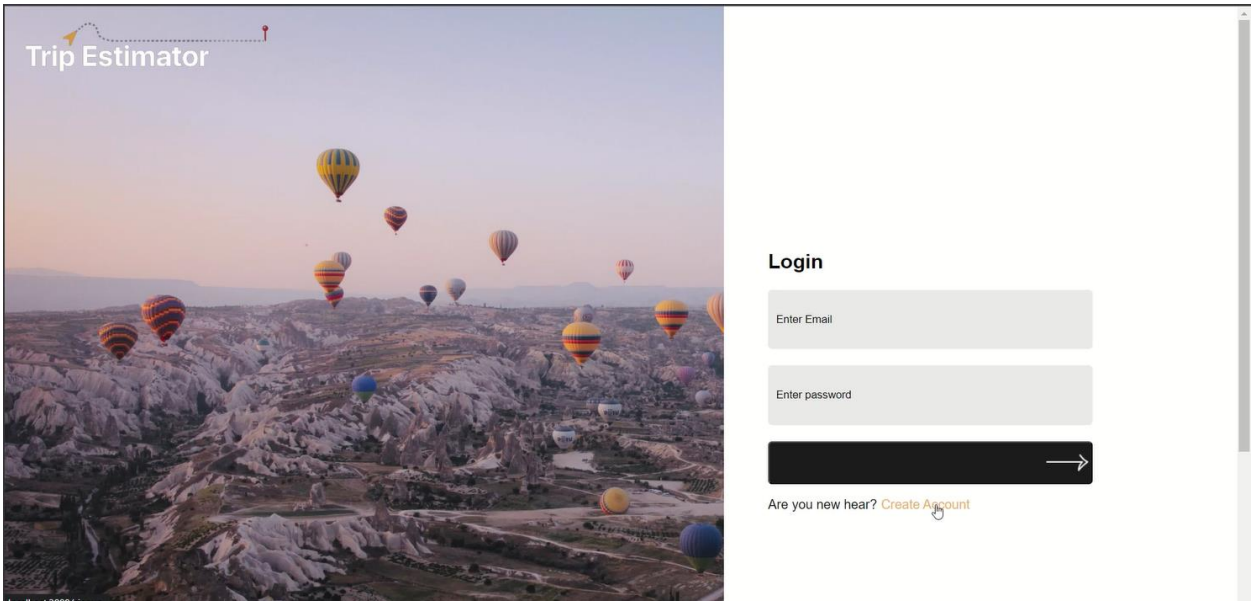

Fig.3.4.Login Page

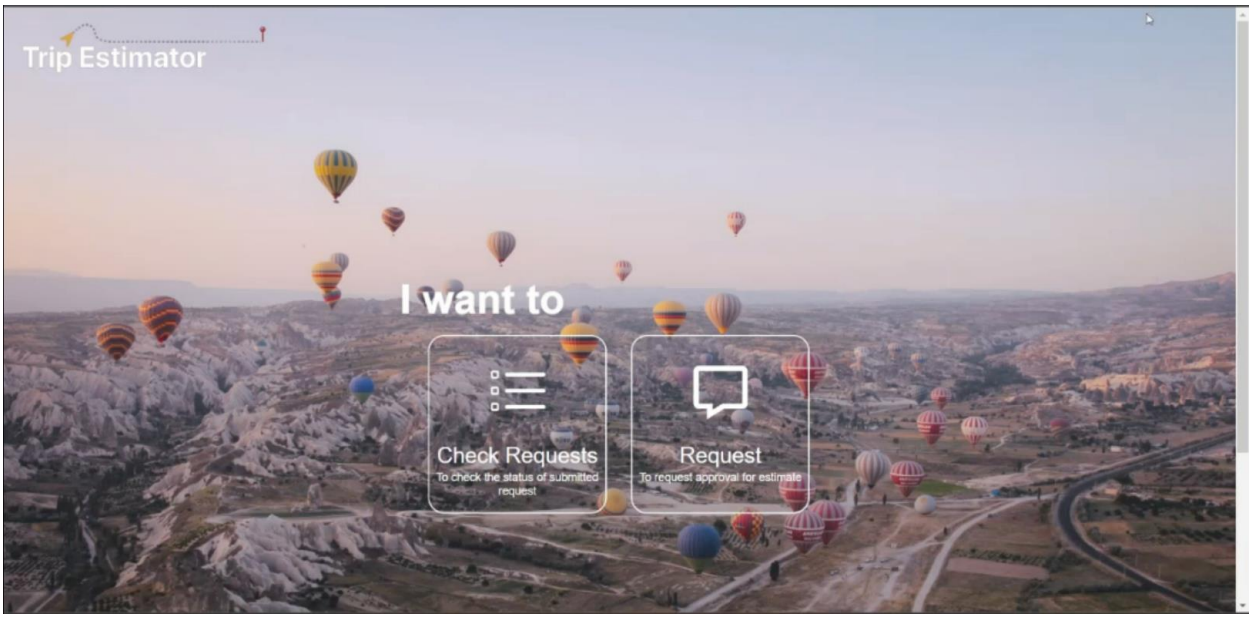

Fig.3.5.Employee Home Page

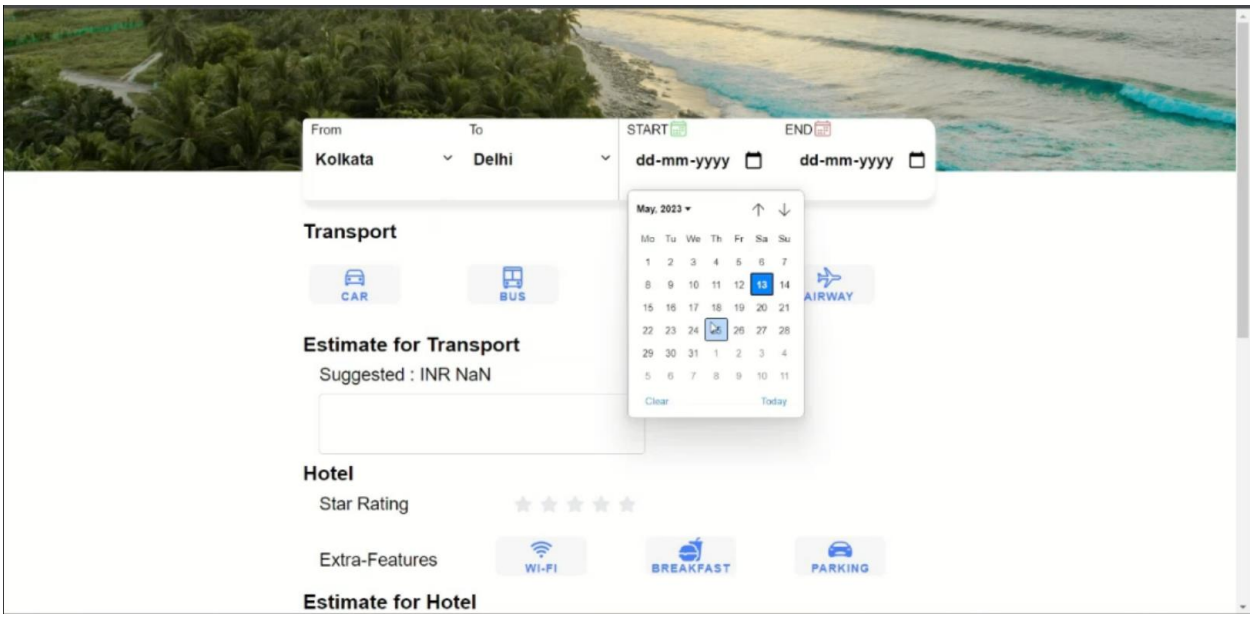

Fig.3.6. Selecting source, destination along with the start and end dates

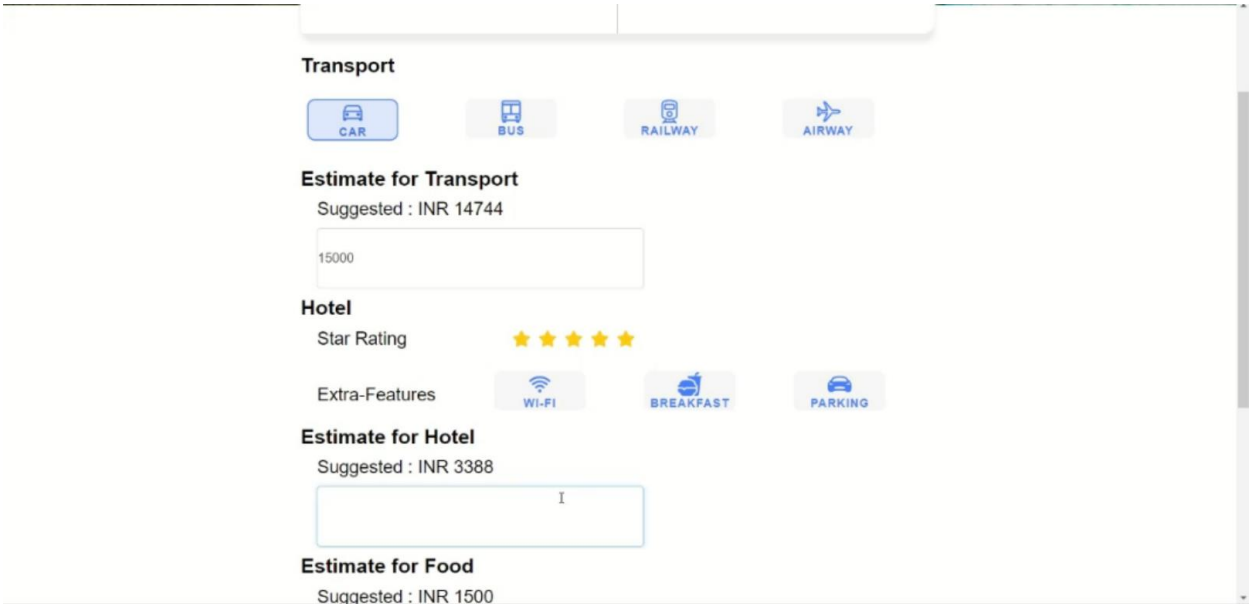

Fig.3.7. Selecting Mode of Transport and Hotel details

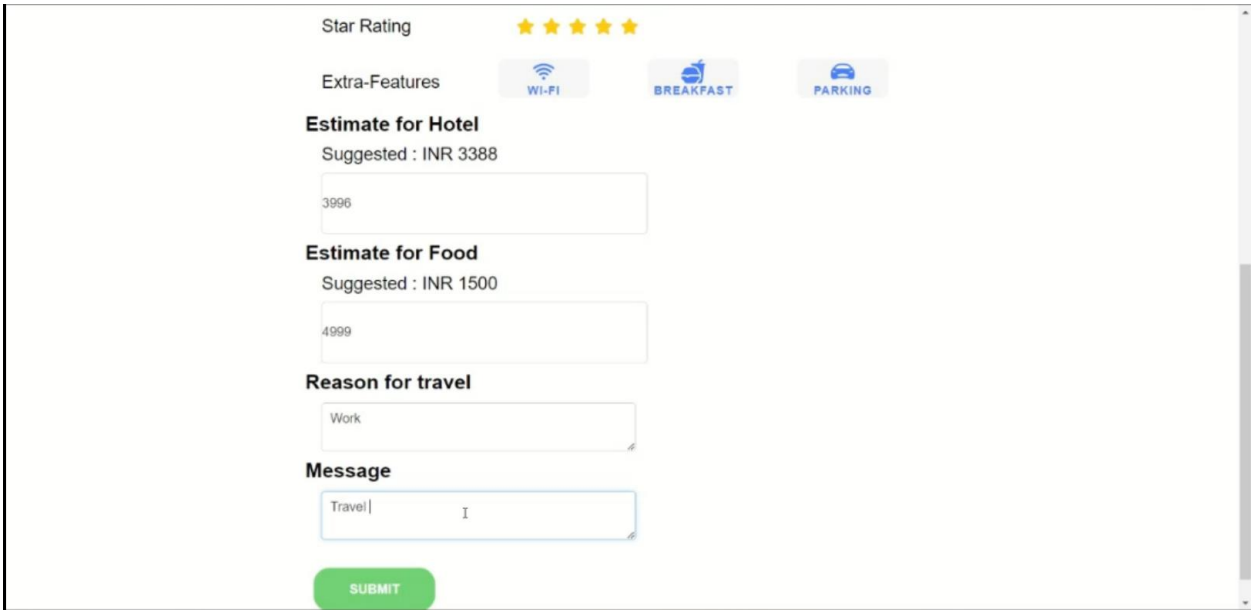

Fig.3.8.Submition of Details

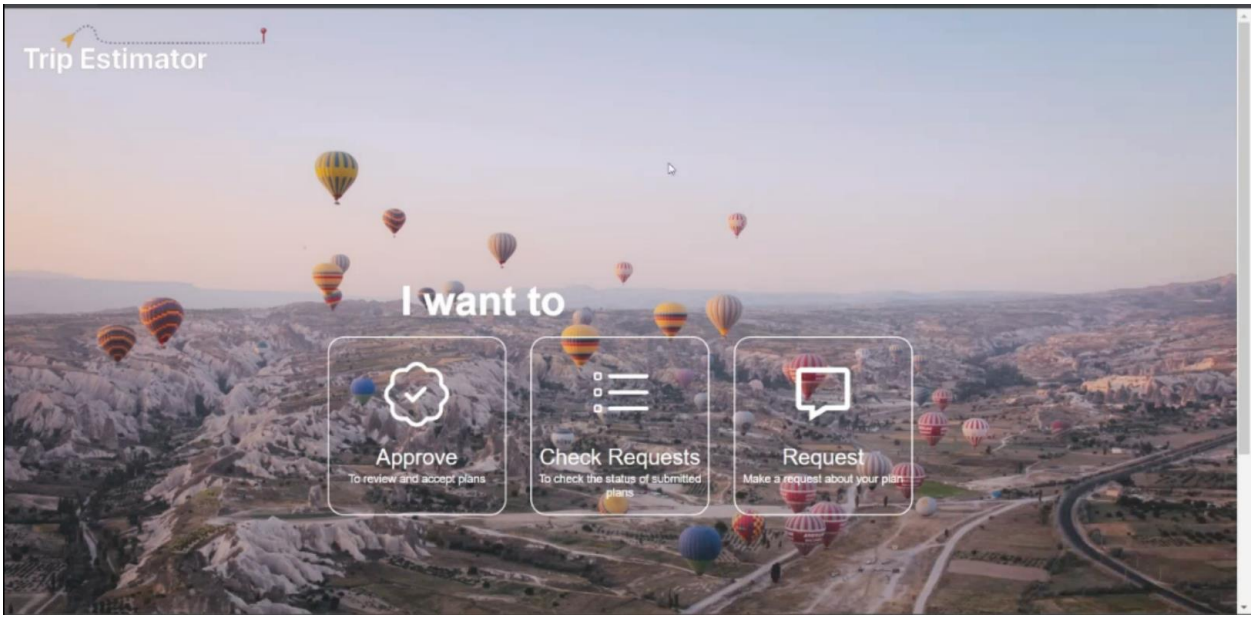

Fig.3.9. Admin Home Page

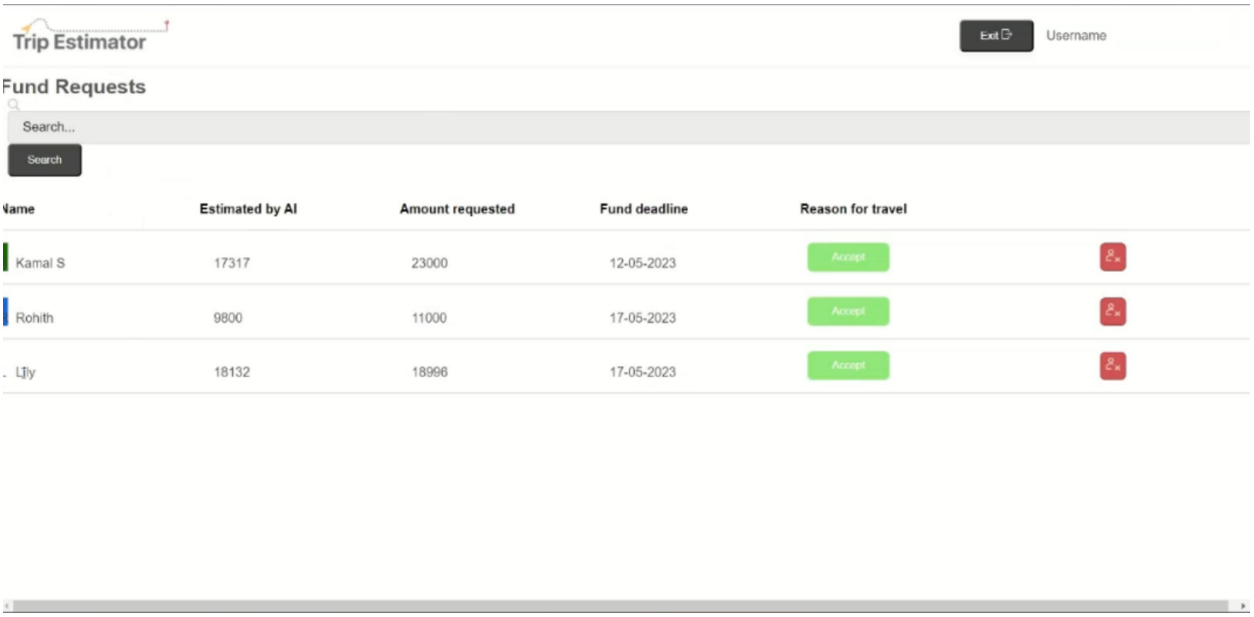

Fig.3.10. Approver's Page wit list of Requests

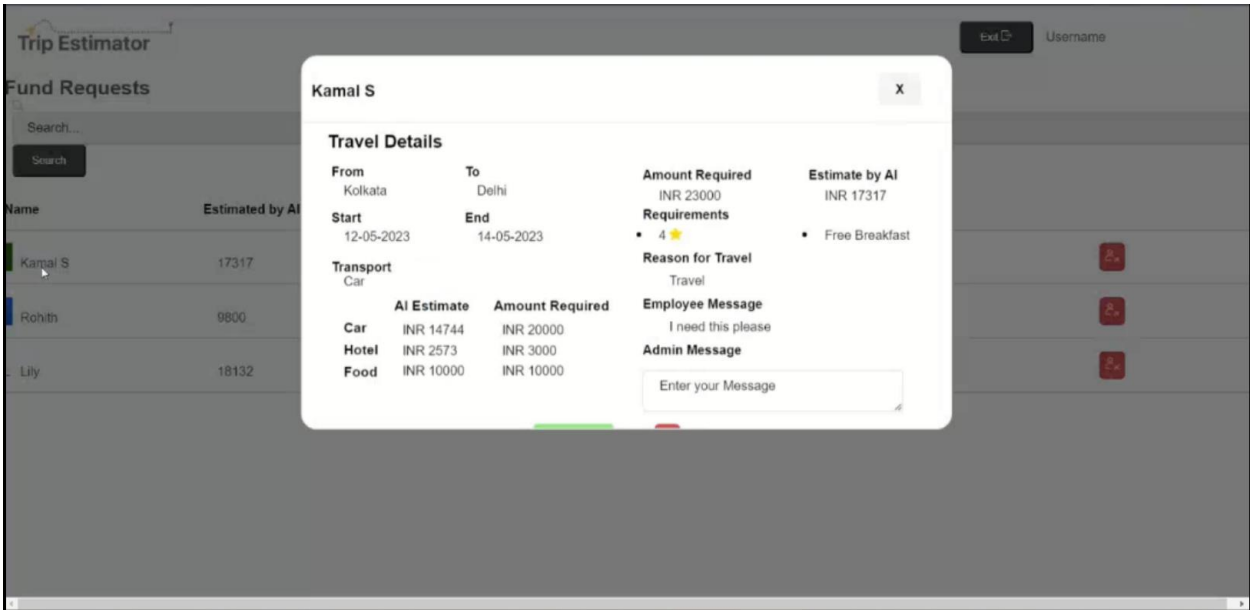

Fig.3.11 Pop-up box for Viewing Additional Details

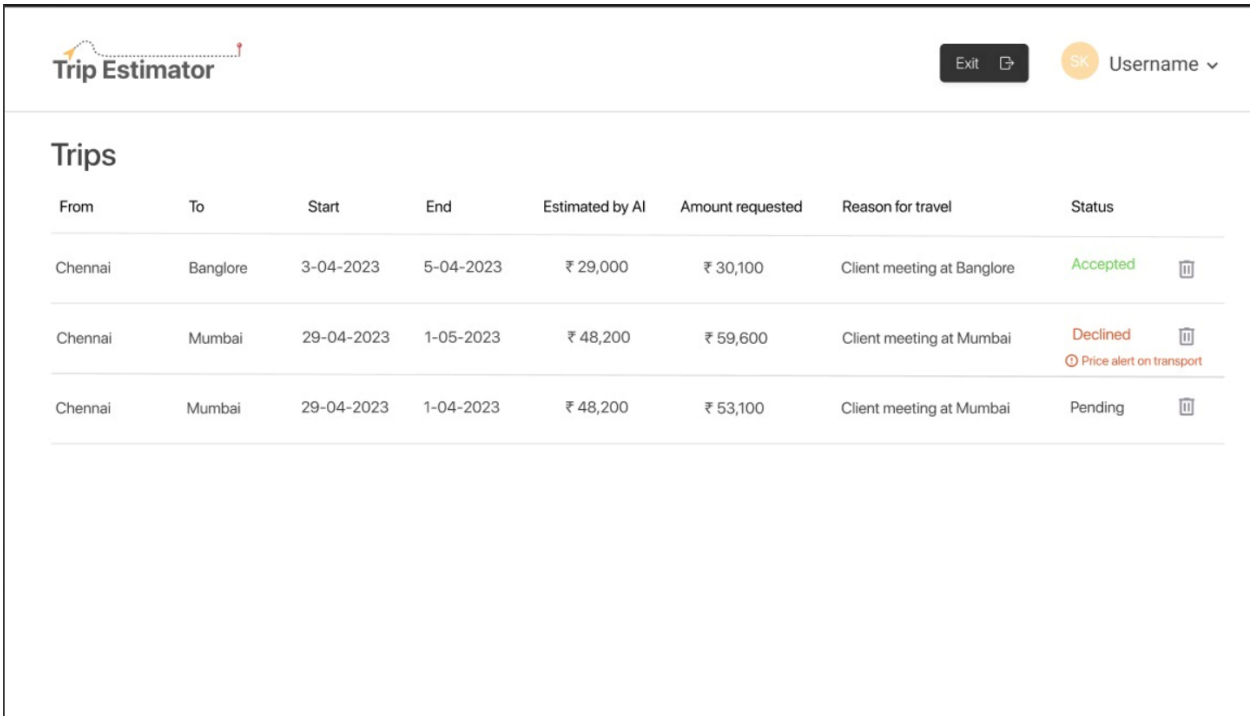

Fig.3.12. Status Page For Employee

3.3 ANALYSIS OF RESULTS

The project titled "Trip Estimation" aimed to develop a robust and accurate system for estimating travel distances and durations. The analysis of the project's results provides valuable insights into the performance and effectiveness of the developed solution. One of the key metrics used to evaluate the trip estimation system was the accuracy of distance predictions. The results showed that the system achieved a high level of accuracy, with the majority of estimated distances falling within a small margin of error. This indicates that the system is capable of providing reliable distance estimates for various types of trips, including short distances within urban areas and long-distance journeys.

Another important aspect of the analysis focused on the estimation of travel durations. The results revealed that the system performed well in predicting the time it takes to complete a trip. However, it was observed that the accuracy of duration estimates varied depending on factors such as traffic conditions and the time of day. In urban areas with heavy congestion, the system tended to underestimate the travel time, while in areas with lighter traffic, the estimates were generally more accurate. This finding suggests that incorporating real-time traffic data could further enhance the system's accuracy in estimating travel durations.

Furthermore, the analysis explored the system's performance across different transportation modes, such as driving, walking, and public transportation. The results indicated that the system excelled in estimating distances and durations for driving trips, as road networks and traffic patterns are relatively well-defined. However, the accuracy of estimates for walking and public transportation trips was slightly lower.

In conclusion, the analysis of the results for the "Trip Estimation" project demonstrates the effectiveness of the developed system in accurately estimating travel distances and durations. While the system performed well overall, there is room for further improvements, particularly in accounting for real-time traffic data and refining the modeling techniques for walking and public transportation trips. The insights gained from this analysis can inform future iterations of the system, leading to a more robust and reliable trip estimation solution for users.

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

CONCLUSION:

In conclusion, this project has achieved the goal of accurately estimating the cost of a trip when the required details of the trip were given as input. The website also provides the facility for a superior to either accept or reject the request made by the employee. The fronted or the user interface provides a simplistic yet aesthetically pleasing appearance for the benefit of the user.

The web-based application involves the process of a person having to create an account and then login, if he happens to be an employee he can enter trip details and make a request, while generating an estimate. And if the person logging in is an admin then he can accept the request or reject it with a message providing the reason for rejection, and what to change about the request.

FUTURE WORK:

The potential for this application in the future is vast. As of now it has been implemented on a small scale covering only the major cities of India, when we get our hands on more data we can build a better algorithm better suited for the new data providing estimates for other cities as destinations. The estimate is also strictly for business trips, and that can be extended to include holiday trips with the inclusion of more details for the prediction and more algorithms with focused prediction patterns providing us with even more accurate estimates for a wide range of trips.

The application can also be included as an extension as part of other estimation projects such as temporary housing or travelling with multiple destinations along the way. The website's can be

REFERENCES

1. Krishna, S. & Sharitha, G. & Ganesh, P. & Kumar, G.V. & Karthika, G.. (2022). Flight Ticket Price Prediction Using Regression Models. International Journal for Research in Applied Science and Engineering Technology. 10. 2084-2089. 10.22214/ijraset.2022.41058.

2. E.A., Kuptsova & S.K., Ramazanov. (2020). Analysis of artificial neural networks training models for airfare price prediction. Artificial Intelligence. 25. 45-50. 10.15407/jai2020.03.045.

3. Kalampokas, Theofanis & Tziridis, Konstantinos & Kalampokas, Nikolaos & Nicolaou, Alexander & Vrochidou, Eleni & Papakostas, George. (2023). A Holistic Approach on Airfare Price Prediction Using Machine Learning Techniques. IEEE Access. PP. 1-1. 10.1109/ACCESS.2023.3274669.

4. Eve Astrid Andersson; Philip Greenspun; Andrew Grumet, "HTML," in Software Engineering for Internet Applications , MIT Press, 2006, pp.329-350.

5. Z. Hao, Z. Limiao and H. Hua, "A Web Design Mode for Browsers to CSS Compatibility Issues," 2012 Fourth International Conference on Multimedia Information Networking and Security, Nanjing, China, 2012, pp. 160-163, doi: 10.1109/MINES.2012.53.

6. A. Javeed, "Performance Optimization Techniques for ReactJS," 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2019, pp. 1-5, doi: 10.1109/ICECCT.2019.8869134.

WEBSITES

1. 1.https://www.w3schools.com

2. 2.https://www.javatpoint.com
3. 3.https://www.youtube.com/@NetNinja
4. 4.https://www.youtube.com/@programmingwithmosh
5. https://flask.palletsprojects.com/en/2.3.x/
6. https://www.geeksforgeeks.org/how-to-connect-reactjs-with-flask-api/
7. https://towardsdatascience.com/build-deploy-a-react-flask-app-47a89a5d17d9
8. https://www.digitalocean.com/community/tutorials/how-to-use-mongodb-in-a-flask-application
9. https://realpython.com/linear-regression-in-python/