Project Report Format

1. INTRODUCTION

1.1 Project Overview

Given the unpredictability surrounding the output of the wind farms, wind power forecasting is crucial in addressing the difficulties of balancing supply and demand in any electrical system. A real-time output power prediction system is crucial for a wind farm that converts wind energy into electrical power. In this project, we use statistical models and physical models to construct a prediction system.

1.2 Purpose

The amount of wind energy used in the world's energy supply is rising. In the recent decades, there has been intense research into the extraction of power from renewable resources in an effort to lessen the global electrical energy crisis and environmental degradation. Because wind power availability cannot be predicted in advance, wind farm operators have trouble planning their systems and energy needs. In order to get over the obstacles, a detailed prognosis is needed. The climate at the location determines how much power a wind farm produces. In this project, we predict the energy output of wind turbines based on weather conditions.

2. LITERATURE SURVEY

2.1 Methodology 1

In Rashid et Al [1], random forest regressor algorithm is used to forecast the output power of the wind turbines. Two years' worth of SCADA data were gathered from a wind farm in France. To anticipate the output power, the wind's direction, speed, and ambient temperature are used as input variables. The paper examined two alternative capacity factors in the model. The estimated mean absolute errors for the proposed model in this study were 3.6% and 7.3% for and 0.2 capacity factors. With a minimum amount of inaccuracy, the proposed model in this study provides an effective way to predict the output power of wind turbines.

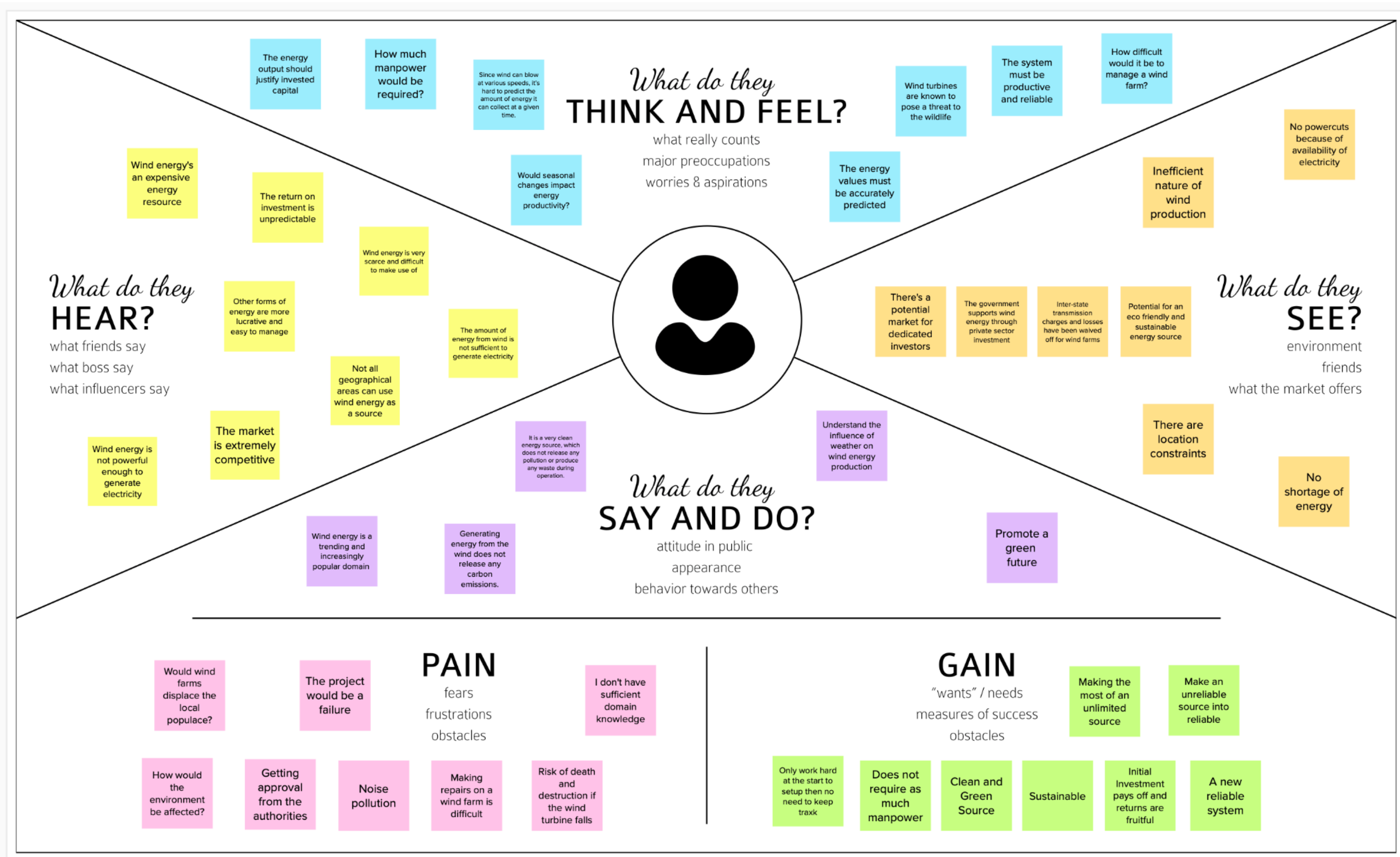
2.2 Methodology 2

[2] states that if the output can be predicted more accurately, look for methods that energy suppliers might better coordinate the cooperative production of multiple energy sources.Soft computing models have the ability to spot trends that can describe a "normal day" in terms of the weather. The data on six meteorological indicators collected in a Spanish city are examined in this multidisciplinary study. Data was gathered in 2007 from a pollution monitoring station that is a part of a network of stations with a similar purpose throughout the Spanish Autonomous Region of Castile-Leon for more than six months. It

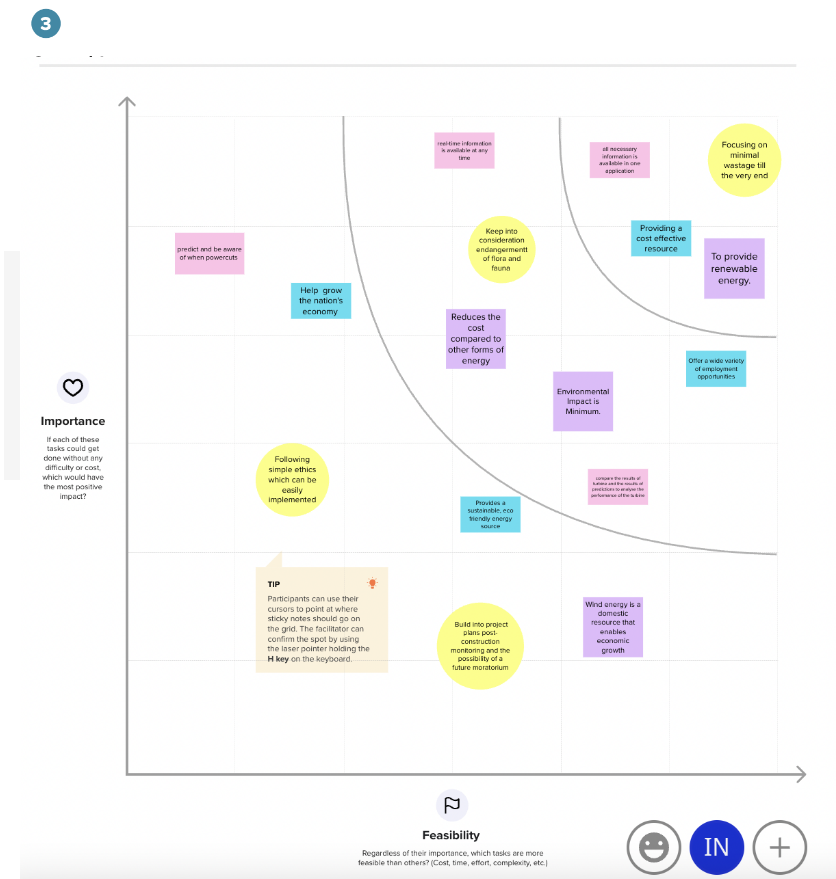
is possible to establish correlations between the meteorological variables and the days of the year by comparing the meteorological data. The use of appropriate data processing methods to analyze meteorological variables and aerosol pollutants to determine typical days is one of the study's major accomplishments.

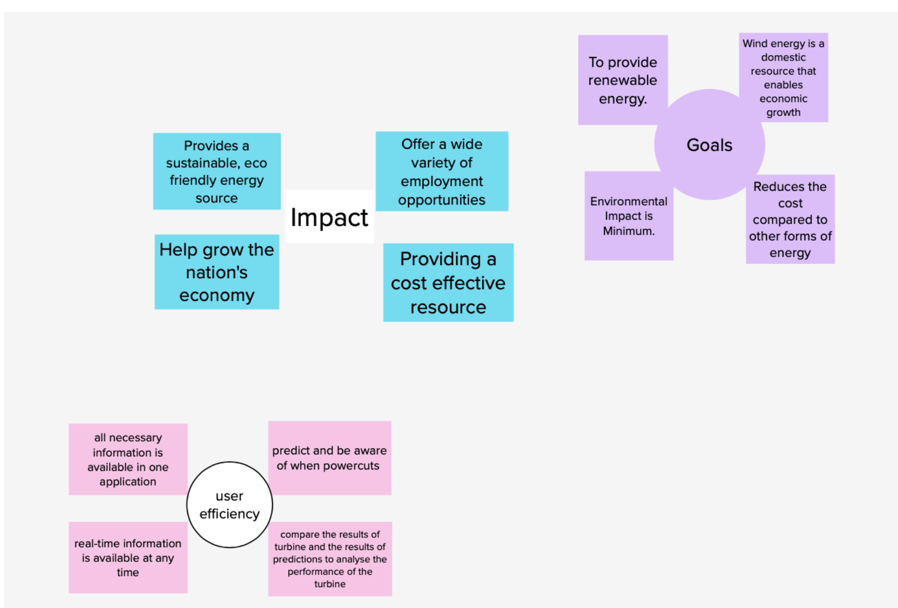
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming





3.3 Proposed Solution

1	Problem Statement (Problem to be solved)	Because wind power availability cannot be predicted in advance, wind farm operators have trouble planning their systems and energy needs. In order to get over the obstacles, a detailed prognosis is needed. The climate at the location determines how much power a wind farm produces. In this project, we predict the energy output of wind turbines based on weather conditions.
2	Idea / Solution description	Use a machine learning model to make accurate predictions of the wind turbine output based on climate using the available information.
3	Novelty / Uniquenes s	· The user interface allows a user to enter the relevant information and obtain an accurate forecast easily. · Data-driven approach to wind farming
4	Social Impact / Customer Satisfaction	· Creating new job opportunities · Gives farmers and ranchers a new stream of income in the form of land lease payments. · Customer satisfaction: With rare exceptions, wind turbines do not emit pollutants that can harm the air or water, and they do not need water for cooling. · Even without technology, windmills have always offered a dependable energy supply.
5	Business Model (Revenue Model)	The ability to predict the output of a wind turbine benefits all the end users. The wind turbine companies will be able to keep track of the performance of their wind turbine, the government will be able to see how much electricity can be obtained from the wind turbines.
6	Scalability of the Solution	· Cloud based hosting could ensure zero down time · As an alternative to traditional relational data storage mechanisms, nosql could be considered to deal with large volumes of data

3.4 Problem Solution Fit

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**User Registration**	Users may create accounts and login using their credentials to use the application. This prevents unauthorized access and keeps their data secure.
FR-2	**User Confirmation**	Confirmation via Email Confirmation via OTP
FR-3	**Dashboard**	Once logged in, the user is redirected to a dashboard wherein the various features provided by the application are made available.
FR-4	**Data Manipulation**	Users have the facility to enter, update and modify their personal details.

FR-5	**Weather Details Display**	Integration with OpenWeatherMap to display forecasts for the next few days
FR-6	**Power Output Prediction**	Getting relevant inputs from the user Communication with the flask backend, and displaying the results
FR-7	**Productivity Stats**	Users can visualize energy output statistics over time in order to gain insights and better understand trends.

4.2 Non-Functional requirements

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

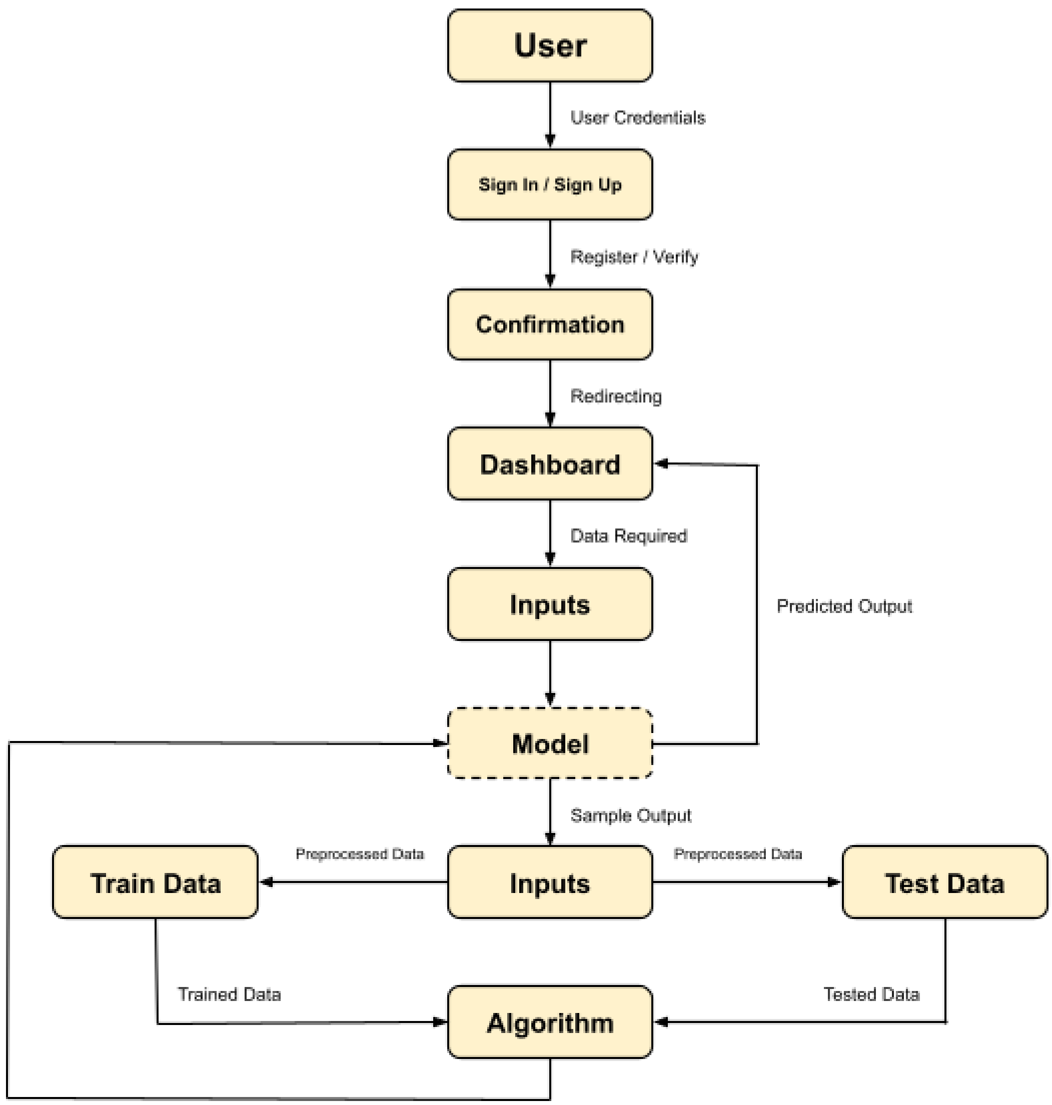
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The User Interface must be simple, elegant and cater to the needs of people with varying degrees of digital literacy.
NFR-2	**Security**	The user's information must be confidential, and the software must be developed keeping common security vulnerabilities in mind.
NFR-3	**Reliability**	The results of the predictive model must be reliable and give the user a clear estimate to work with.
NFR-4	**Performance**	The predictions must be returned with minimal latency.
NFR-5	**Availability**	Hosting the application on IBM cloud ensures zero to little down time, thus making it available around the clock.

NFR-6	**Scalability**	The application must display horizontal as well as vertical scalability, adapting to a larger user base while being open to the addition of new features.

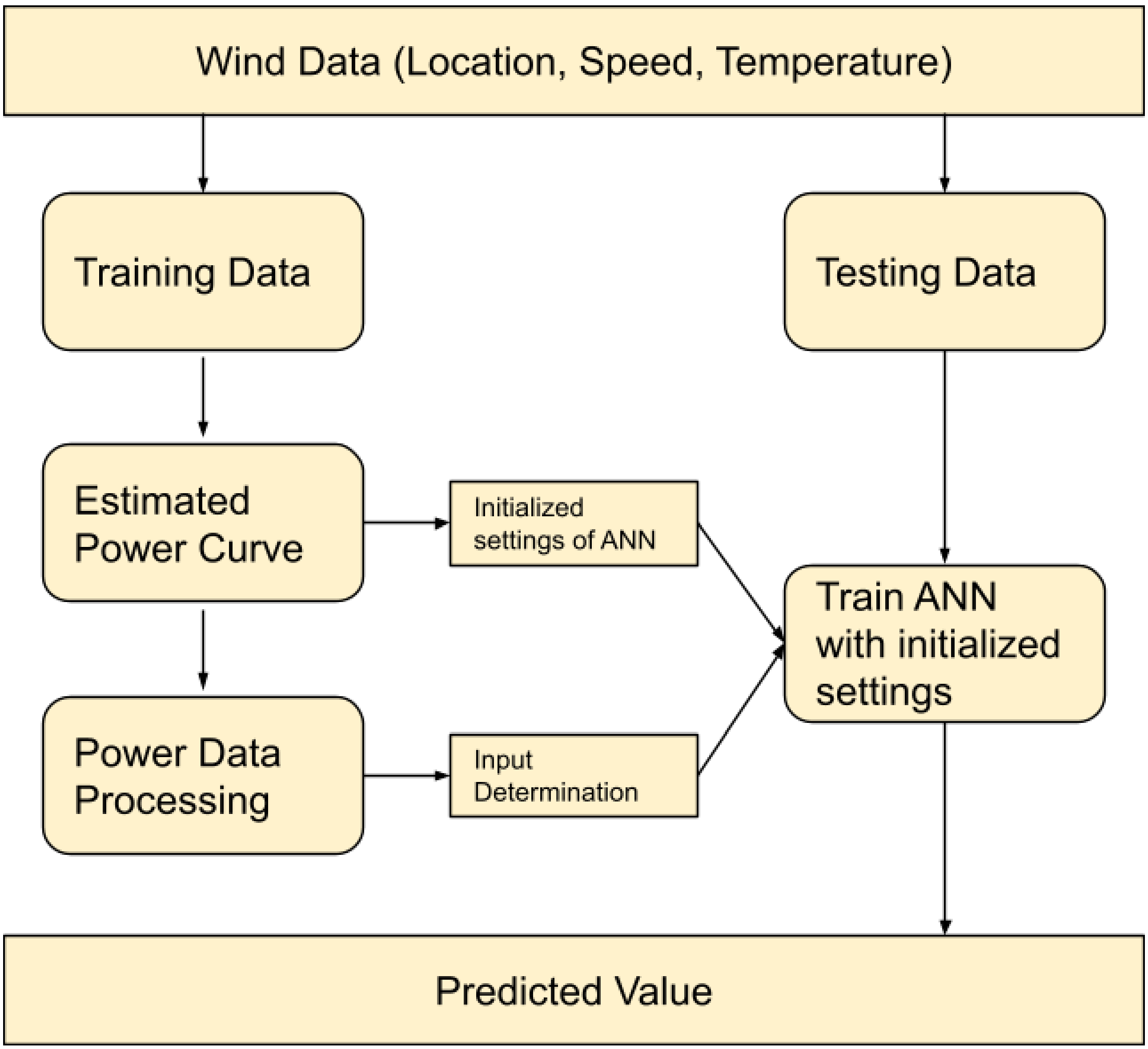
5. PROJECT DESIGN

5.1 Data Flow Diagrams

Level 0

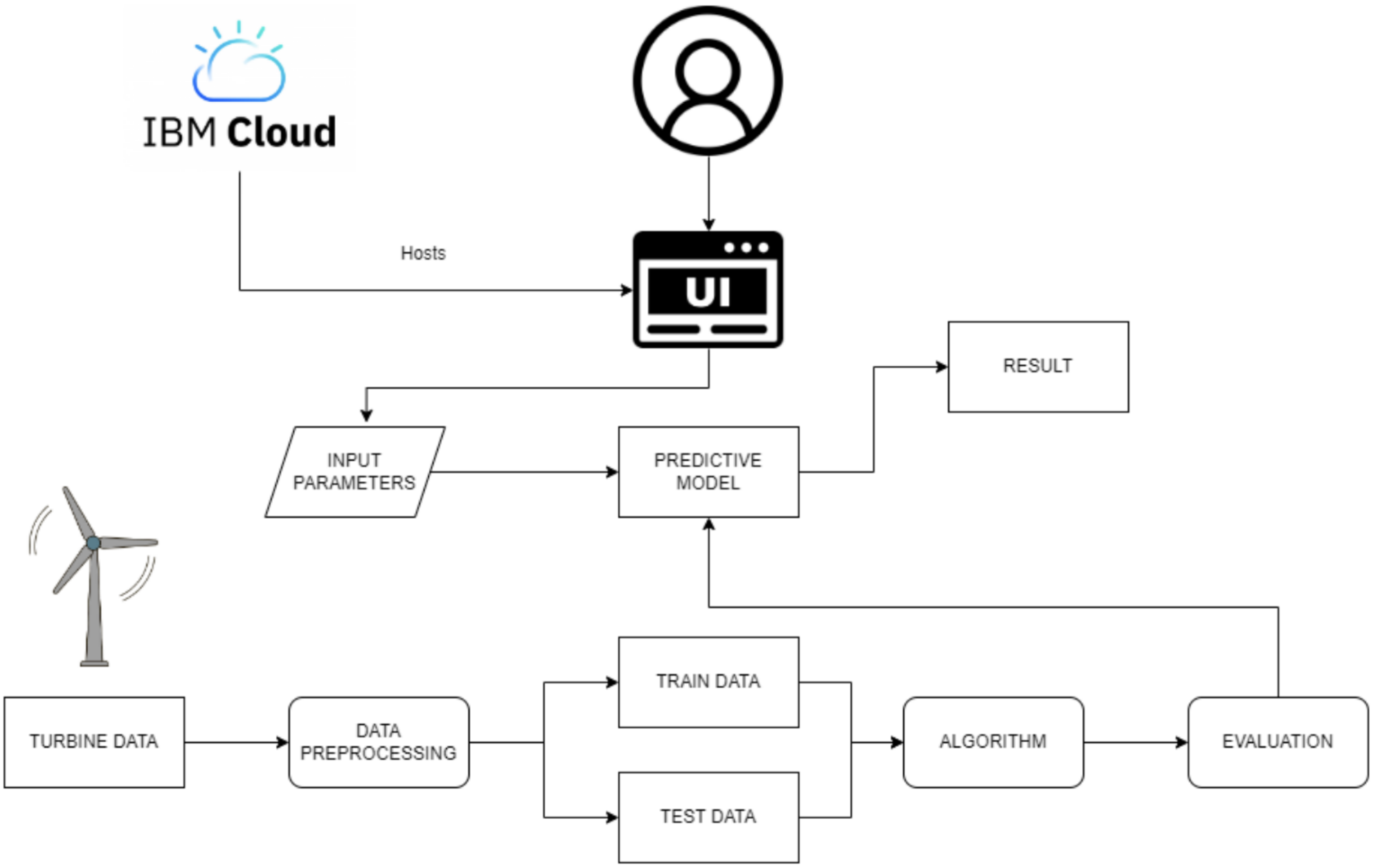


Level 1 - Predictive Model



5.2 Solution Architecture

Solution Architecture:



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my details like email and password	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by using my email & password	I can access the dashboard	High	Sprint-1
	Dashboard	USN-4	As a user, I can get information about wind energy. If I press the predict energy button, I can enter the input details and get the prediction	I can predict for single sample	High	Sprint-1
		USN-6	As a user, I can get visual representation of the prediction	I can have single output	High	Sprint-1
		USN-7	As a user, I can view the detailed report of my prediction	I can access details of my process and prediction	Medium	Sprint-1
Customer Care Executive	Documentation	USN-8	As a helper, I can refer the documentation for support and guidance	I can use user manual for guidance	Medium	Sprint-1,2,3,4
Administrator	Settings	USN-9	As a developer, I can access dashboard's settings and view the API token	I can view the API token for creating request	Low	Sprint-4
	Feedback	USN-10	As a developer, I am able to view user's feedback	I can customize these web page based on feedback	Medium	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Product Backlogs, Sprint Schedule and Estimation

Sprint	Functional Requirement (EPIC)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint 1	Registration	USN-1	I can sign up for the application as a user by providing my email address, password, and password confirmation.	5	High	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 1		USN-2	When I register for the application as a user, I will get a confirmation email.	4	High	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 1		USN-3	I can sign up for an application as a user using my phone number.	4	Low	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 1		USN-4	I can sign up for the application as a user using Gmail.	3	Medium	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 1	Login(User)	USN-5	I can login to the application as a user by providing my email and password.	5	High	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G

Sprint 2	Dashboard	USN-6	After logging in, I can access my dashboard.	6	Medium	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 2	Web access	USN-7	As a customer, I can use the website to predict the weather conditions.	7	High	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 2	Prediction	USN-8	As a customer, when I provide the weather data, the website should forecast the approximate weather conditions.	7	High	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 3	Analysis	USN -9	I want to analyze and store my predictions as a customer.	10	Medium	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 3	Security	USN-10	As a customer, I anticipate that my data will be secure	10	Medium	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G
Sprint 4	Database Access	USN- 11	I should maintain the website as an administrator. and frequently update the website.	20	Low	Pranay Varma Hemalatha M Isha A.K. Nanda Rochana G

6.2 Sprint Delivery Schedule

Sprint	Total story points	Duration	Sprint start date	Sprint end date (Planned)	Story points completed (as on planned End date)	Sprint Release date(Actual)
Sprint 1	20	6 days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint 2	20	6 days	31 Oct 2022	5 Nov 2022	20	5 Nov 2022
Sprint 3	20	6 days	7 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint 4	20	6 days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

7. CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

ANN pretrained model deployed on IBM Watson.

We use the Watson Machine Learning Python client library or the Watson Machine Learning API to create, train, and deploy models directly from notebooks. Structured data is automatically preprocessed by AutoAI, which then chooses the ideal estimator for the situation and creates model candidate pipelines for us to analyze and contrast. We can create a machine learning model using the pipeline that performs the best. We can then run experiments in Experiment Builder to train complex models and then deploy our models so that we can evaluate them and generate predictions.

7.2.1 Importing necessary Libraries

```python
#import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from collections import Counter
import seaborn as sns
import os
```

7.2.2 Data Formatting

In [6]:
```python
#renaming the columns to our convention
data.rename(columns = {'LV ActivePower (kW)':'ActivePower(kW)',
                       "Wind Speed (m/s)":"WindSpeed(m/s)",
                       "Wind Direction (°)":"WindDirection","Theoretical_Power_Curve (KWh)":"TheoreticalPowerCurve(KWh)"},
            inplace = True)
data.head()
```

Out[6]:

	Date/Time	ActivePower(kW)	WindSpeed(m/s)	TheoreticalPowerCurve(KWh)	WindDirection
0	01 01 2018 00:00	380.047791	5.311336	416.328908	259.994904
1	01 01 2018 00:10	453.769196	5.672167	519.917511	268.641113
2	01 01 2018 00:20	306.376587	5.216037	390.900016	272.564789
3	01 01 2018 00:30	419.645904	5.659674	516.127569	271.258087
4	01 01 2018 00:40	380.650696	5.577941	491.702972	265.674286

In [7]:
```python
#data formatting
data['Date/Time'] = pd.to_datetime(data['Date/Time'],format='%d %m %Y %H:%M')
data['year'] = data['Date/Time'].dt.year
data['month'] = data['Date/Time'].dt.month
data['day'] = data['Date/Time'].dt.day
```

7.2.3 Calculating Average Values

In [9]:
```python
#finiding the mean speed
def mean_speed(x):
    x = round(x,2)
    a = x//1
    a,b = a+0.25,a+0.75
    if x < a:
        x = a - 0.25
    else:
        x = b -0.25
    return x
```

In [10]:
```python
data['meanSpeed'] = data['WindSpeed(m/s)'].apply(mean_speed)
data.head(100)
```

Out[10]:

	Date/Time	ActivePower(kW)	WindSpeed(m/s)	TheoreticalPowerCurve(KWh)	WindDirection	year	month	day	Hour	minute	meanSpeed
0	2018-01-01 00:00:00	380.047791	5.311336	416.328908	259.994904	2018	1	1	0	0	5.5
1	2018-01-01 00:10:00	453.769196	5.672167	519.917511	268.641113	2018	1	1	0	10	5.5
2	2018-01-01 00:20:00	306.376587	5.216037	390.900016	272.564789	2018	1	1	0	20	5.0
3	2018-01-01 00:30:00	419.645904	5.659674	516.127569	271.258087	2018	1	1	0	30	5.5
4	2018-01-01 00:40:00	380.650696	5.577941	491.702972	265.674286	2018	1	1	0	40	5.5
...	...	...	...	...	...	...	...	...	...	...	...
95	2018-01-01 15:50:00	2820.512939	10.772420	3186.029883	225.276398	2018	1	1	15	50	10.5
96	2018-01-01 16:00:00	2812.279053	10.647520	3133.259224	224.680603	2018	1	1	16	0	10.5
97	2018-01-01 16:10:00	2530.447021	9.982661	2781.274041	225.519501	2018	1	1	16	10	9.5
98	2018-01-01 16:20:00	2399.121094	9.874386	2711.492458	227.273804	2018	1	1	16	20	9.5
99	2018-01-01 16:30:00	2335.587891	9.785480	2651.341009	229.255493	2018	1	1	16	30	9.5

7.2.4 Plotting a graph for theoretical power curve vs actual power curve

```
In [18]: def graph_T(i):
    fig = plt.figure(figsize=(20,10))
    plt.plot(list_table[i]["WindSpeed(m/s)"],
             list_table[i]["TheoreticalPowerCurve(KWh)"],
             label = "Theoretical Power Curve",
             marker = "o", markersize = 10, linewidth = 5)

    plt.plot(list_table[i]["WindSpeed(m/s)"],
             list_table[i]["ActivePower(kW)"],
             label = "Actual Power Curve",
             marker = "o", markersize = 10, linewidth = 5)

    plt.xlabel("Wind Speed (m/s)")
    plt.ylabel("Power (kW)")
    plt.title("Direction towards {}".format(list_tableName[i]))
    plt.legend()
    plt.show()
    fig.savefig("{}_Powercurve.jpeg".format(list_tableName[i]))
    plt.close(fig)
```

```
In [19]: graph_T(0)
```

7.2.5 Splitting the testing and training set

```
In [22]: from sklearn.model_selection import train_test_split
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_squared_error , r2_score
import joblib
```

```
In [23]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 42)
```

7.2.6 Implementing model and making prediction, and evaluation of performance

```
In [24]: lasso = Lasso(alpha = 0.01)
model = lasso.fit(X_train, y_train)
pred_train_lasso= lasso.predict(X_train)

print("Training RMSE and R2 score:")
print(np.sqrt(mean_squared_error(y_train,pred_train_lasso)))
print(r2_score(y_train, pred_train_lasso))

pred_test_lasso= lasso.predict(X_test)
print("Testing RMSE and R2 score:")
print(np.sqrt(mean_squared_error(y_test,pred_test_lasso)))
print(r2_score(y_test, pred_test_lasso))
```

```
Training RMSE and R2 score:
533.882754117229
0.8349081611466808
Testing RMSE and R2 score:
539.8257889459195
0.8292153379339346
```

7.2.7 Deployment of ML MODEL in IBM WATSON

```
In [122]: !pip install ibm_watson_machine_learning
```

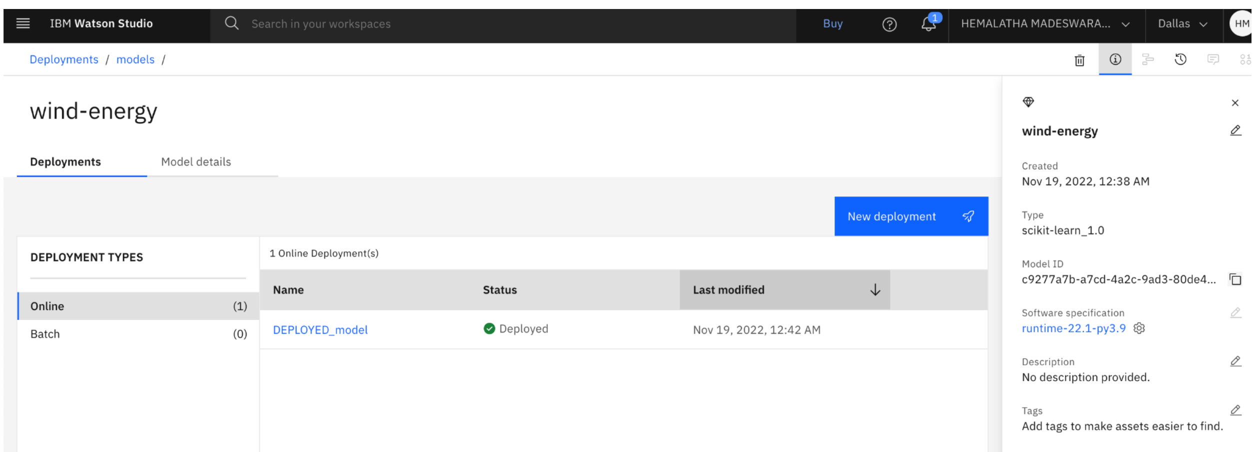
```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.3.4)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (4.8.2)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_m
ine_learning) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_l
ning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine
arning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sd
```

7.2.8 Connection of jupyter notebook to IBM WATSON

```
(3.0.4)

In [123]: from ibm_watson_machine_learning import APIClient
          wml_credentials = {
                              "url": "https://us-south.ml.cloud.ibm.com",
                              "apikey":"ZDs-CVs3SY4Oc91J9avZDK7TXRNeWZd4sErddu5Yfex1"
          }
          client = APIClient(wml_credentials)

In [124]: #creating a deployment space
          def guide_from_space_name(client, space_name):
              space = client.spaces.get_details()
              return(next(item for item in space['resources'] if item['entity']["name"] == space_name)['metadata']["id"])

In [125]: space_uid = guide_from_space_name(client, 'models')
          print("space uid = "+space_uid)

            space uid = 6e79ef45-9da4-4e87-a246-b6d7aa928dfb
```

7.2.9 Adding Model in IBM WATSON

```
In [147]: software_spec_uid = client.software_specifications.get_uid_by_name("runtime-22.1-py3.9")
          software_spec_uid

Out[147]: '12b83a17-24d8-5082-900f-0ab31fbfd3cb'

In [151]: model_details = client.repository.store_model(model=lasso,meta_props={
          client.repository.ModelMetaNames.NAME:"wind-energy",
          client.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
          client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid},training_data=X_train,training_target=y_train )

          model_id = client.repository.get_model_id(model_details)

In [150]: model_id

Out[150]: '7963cd8b-2969-459d-a8b5-c72727b4554a'
```



7.2 Feature 2

Web Application developed using HTML and css with a flask backend.

Features - Sign up/Sign in

Dashboard for current weather details and wind energy prediction

Statistics page for wind energy production in India

APIs used - OpenWeatherMap for current weather and 4 from data.gov.in for wind energy production details

DB used - sqlite

7.3 Database Schema (if Applicable)

8. TESTING

8.1 Test Cases

				Date	18-Nov-22								
				Team ID	PNT2022TMID35585								
				Project Name	Predicting The Energy Output of Wind Turbine Based on Weather Conditions								
				Maximum Marks	4 marks								
Test case ID	**Feature Type**	**Component**	**Test Scenario**	**Pre-Requisite**	**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**	**Status**	**Comments**	**TC for Automation(Y/N)**	**BUG ID**	**Executed By**
LoginPage_TC_OO1	Functional	Login page	Verify user is able to log in with valid credentials		1. Enter email ID 2. Enter Correct Password 3.Press 'login' or enter key.	Email ID : pranayv01@gmail.com Password: abcd	Home Page must be seen	Working as Expected	Pass	Test Case Successfully Passed	N		
LoginPage_TC_OO2	Functional	Login page	Verify user is able to log into application with Invalid credentials		1. Enter email ID 2. Enter Incorrect Password 3.Press 'login' or enter key.	Email ID : pranayv01@gmail.com Password: abbd	Please check your login details and try again' must be seen above the login box	Working as Expected	Pass	Test Case Successfully Passed	N		
SignupPage_TC_OO1	Functional	Signup Page	Verify user is able to create an account		1.Enter email ID 2.Enter Password 3.Press 'Signup' or enter key.	Username: chalam@gmail.com password: Testing123	User should be redirected to login page	Working as Expected	Pass	Test Case Successfully Passed	N		
WeatherConditions_TC_OO1	Functional	Home Page	Verify user is able to see the current weather conditions for the entered city		1.Login 2. Enter city name in the input field for the weather conditions 3. Click on 'Go' or press the enter key	City Name : Chennai	The current weather conditions for Chennai should be displayed in a box below the input field	Working as Expected	Pass	Test Case Successfully Passed	N		
WeatherConditions_TC_OO2	UI	Home Page	Verify user is able to see the current weather conditions for a wrongly entered city name		1.Login 2.Enter a wrong city name in the input field for the weather conditions 3.Click on 'Go' or press the enter key	City Name : Chenai	The message "oops, not found" must be displayed below the input box	Working as Expected	Pass	Test Case Successfully Passed	N		
EnergyPrediction_TC_OO1	Functional	Home Page	Verify user is able to obtain wind energy prediction if all parameters are entered correctly		1.Login 2.Enter Wind Direction, Wind Speed, Hour,Day and Month in the Wind Prediction form	Wind Speed : 4 Wind Direction : 245 Hour : 4 Day : 6 Month:5	The predicted wind energy generated must be displayed in a box below the form	Working as Expected	Pass	Test Case Successfully Passed	N		
EnergyPrediction_TC_OO2	UI	Home Page	Verify user is able to obtain wind energy prediction if not all parameters are entered.		1.Login 2.Enter Wind Direction Wind Speed, Hour,Day and Month in the Wind Prediction form	Wind Speed : 4 Wind Direction : 245 Hour : 4 Day : 6 Month:	The message "Please Fill in Every Field" must be displayed below the input box	Working as Expected	Pass	Test Case Successfully Passed	N		
Stats_TC_001	UI	Statistics Page	Verify the statistics page appears when the corresponding link is clicked		1.Login 2.Click on the statistics Link in the top navbar		The statistics page must be rendered with a few buttons seen	Working as Expected	Pass	Test Case Successfully Passed	N		
Stats_TC_002	UI + Functional	Statistics Page	Verify the different visualizations appear when their buttons are clicked		1.Login 2. Click on the statistics link in the top navbar. 3. Click each of the buttons seen		Graphs must be appropriately rendered depending on which button has been clicked	Working as EXpected	Pass	Test Case Successfully Passed	N		
Logout_TC_001	Functional	Login page	Verify the user logs out successfully on clicking the logout button		1.Login 2.Click on the 'Logout' Button in the top navbar		The user must be redirected to the login page	Working as Expected	Pass	Test Case Successfully Passed	N		
Auth_TC_001	Functional	Home Page	Verify the user is able to get weather conditions without logging in		1.Open the application 2.Enter City Name in the input field of the weather conditions form 3.Select 'Go'		The user must be redirected to the login page with a message 'Please log in to access this page.' on top of the login box	Working as Expected	Pass	Test Case Successfully Passed	N		
Auth_TC_002	Functional	Home Page	Verify the user is able to get wind energy prediction without logging in		1.Open the application 2.Fill in the wind energy prediction form 3.Select 'Predict'		The user must be redirected to the login page with a message 'Please log in to access this page.' on top of the login box	Working as Expected	Pass	Test Case Successfully Passed	N		

8.2 User Acceptance Testing

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	0	1	2	0	3
Duplicate	0	0	0	0	0
External	0	0	0	0	0
Fixed	0	1	2	0	3
Not Reproduced	0	0	0	0	0
Skipped	0	0	0	0	0
Won't Fix	0	0	0	0	0
Totals	0	2	4	0	6

Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	7	0	0	7

Security	2	0	0	2
Outsource Shipping	0	0	0	0
Exception Reporting	2	0	0	2
Final Report Output	11	0	0	11
Version Control	1	0	0	1

9. RESULTS

9.1 Performance Metrics

9.1.1 MACHINE LEARNING MODEL:

Since the model is a regression model, it was evaluated against the following metrics:

1. Root Mean Square Error (RMSE) - 539.825788945919

2. R^2 - 0.8349081611466808

CONCLUSION

In this project, we are able to predict the energy output of wind turbines based on the weather conditions. With our model's help, farmers and ranchers benefit the most. Not only does it help in creating new job opportunities, it also allows wind farm operators to plan their systems and energy needs.

REFERENCES

[1] Rashid, Haroon, Waqar Haider, and Canras Batunlu. "Forecasting of wind turbine output power using machine learning." 2020 10th International Conference on Advanced Computer Information Technologies (ACIT). IEEE, 2020.

[2] Corchado, Emilio, Angel Arroyo, and Verónica Tricio. "Soft computing models to identify typical meteorological days." Logic Journal of the IGPL 19.2 (2011): 373-383.

GitHub & Project Demo Link

Github - https://github.com/IBM-EPBL/IBM-Project-54847-1662539698

Demo -
https://drive.google.com/file/d/1U6xMDqmJV9_pXjAYflf2PZsqC7J-twqk/view?usp=sharing

This repository contains the demo link along with all source files