

PREDICTING THE ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER CONDITION

A PROJECT REPORT

Submitted by

NAME	REGISTER NO
S.R. VIDHYAMBIKA (TEAM LEAD)	710719104108
A. JEEVIKA	710719104301
R.K. NAVINA	710719104068
S. PRIYADARSHINI	710719104075

TEAM ID: PNT2022TMID31390

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Wind power generation differs from conventional thermal generation due to the stochastic nature of wind. Thus, wind power forecasting plays a key role in dealing with the challenges of balancing supply and demand in any electricity system, given the uncertainty associated with the wind farm power output. Accurate wind power forecasting reduces the need for additional balancing energy and reserve power to integrate wind power. For a wind farm that converts wind energy into electrical power, a real-time prediction system of the output power is significant. In this project titled "Predicting the energy output of wind turbine based on weather condition," a machine learning algorithm name Random Forest Regressor has been used which will reduce the uncertainty in the prediction of wind power generation so as to meet the generation and demand. This machine learning algorithm is deployed in IBM Watson as well as in flask as a web app. The dataset collected for a period of two years is taken and used for training and testing the ml model. The dataset contains weather parameters like Temperature, Humidity, & pressure and wind parameters like wind speed and wind direction at different heights i.e at 10m, 30m, 50m, and at the height of the wind turbine. To make live prediction Height of the windmill and city name are obtained from the user. Then, based on the city name using Open weather API we can get the weather and wind parameters. With these parameters the prediction is made and the results achieved are displayed in the uploaded video.

It can be observed from the results that the wind energy output predicted values are more accurate and will be more useful in the scheduling of power generation by any wind power producer and also to the power utility. (Eg. State Transmission Utility like TNEB / TANGEDCO Ltd.)

1.2 PURPOSE

Meeting out the power generation and demand and ease of power management to the power utility like TNEB.

CHAPTER 2

LITERATURE SURVEY

[1] FORECASTING OF WIND TURBINE OUTPUT POWER USING MACHINE LEARNING

In this paper, the output power of the wind turbines was predicted using the random forest regressor algorithm. The SCADA data was collected for two years from a wind farm located in France. The model was trained using the data from 2017. The wind direction, wind speed and outdoor temperature were used as input parameters to predict output power. Then, the model was tested for two different capacity factors. The estimated mean absolute errors for the proposed model in this study were 3.6% and 7.3% for 0.2 and 0.4 capacity factors, respectively.

Objective:

The main objective of this paper was to offer an efficient method to predict the output power of wind turbine with preferably low error.

Advantages:

- Real-Time data collected and used as a dataset.
- High accuracy.

Disadvantages:

- Climate varies from year to year due to external factors like global warming, seasonal changes.
- Day to day weather and location are not taken as parameter for prediction.

[2] GREY PREDICTORS FOR HOURLY WIND SPEED AND POWER FORECASTING

This paper uses grey predictor models such as Traditional Grey Rolling Model GM(1,1), Improved Shifted Grey Model, Averaged Grey Model for forecasting for 1 hour ahead average hourly wind speed. Forecasting using the traditional GM(1,1) model revealed an improvement over the persistent model. But, this model is characterized by the occurrence of some overshoots in the predicted time series. Such overshoots can result in predicting wind speed time series worse than that of the persistent model. To overcome the problem of overshoots occurrence, three modified versions of the GM(1,1) model were introduced. The adaptive alpha-based grey model and the improved Grey model achieved higher levels of improvement over the persistent model than the traditional GM(1,1) model. However, those models lack the good tracking characteristic for the actual wind speed time series of the traditional model. The averaged Grey model attained the highest level of wind speed forecasting and wind power prediction accuracy, compared with the persistent model and the

other presented Grey models, while demonstrating a very good tracking feature and a reduction in the overshoot occurrence.

Advantages:

The averaged Grey model attained the highest level of wind speed forecasting and wind power prediction accuracy while demonstrating a very good tracking feature and a reduction in the overshoot occurrence.

Disadvantages:

Does not include problems due to line breakdown, power flow and power quality issues as parameters and generation-load prediction matching is not shown.

[3] USING MACHINE LEARNING TO PREDICT WIND TURBINE POWER OUTPUT

In this paper, new aero structural simulations of a generic 1.5 MW turbine were used to rank atmospheric influences on power output. Simulation data parameters were the hub height wind speed followed by hub height turbulence intensity and then wind speed shear across the rotor disk. These simulation data were used to train regression trees which was implemented in MATLAB and it predicted the turbine response for any combination of wind speed, turbulence intensity, and wind shear that might be expected at a turbine site. For a randomly selected atmospheric condition, the accuracy of the regression tree power predictions was three times higher than that from the traditional power curve methodology. The regression tree method can also be applied to turbine test data and used to predict turbine performance at a new site. The predictions in this study used an ensemble of 100 regression trees. No new data are required in comparison to the data that are usually collected for a wind resource assessment.

Advantages:

Significantly reduce bias in power predictions that arise because of the different turbulence and shear at the new site, compared to the test site.

Disadvantages:

Implementing the method requires turbine manufacturers to create a turbine regression tree model from test site data.

[4] PREDICTING THE ENERGY OUTPUT OF WIND FARMS BASED ON WEATHER DATA: IMPORTANT VARIABLES AND THEIR CORRELATION

This paper proposed energy prediction in a computer science perspective based on weather data and analysed the important parameters as well as their correlation on the energy output. To deal with the interaction of the different

parameters, symbolic regression was used based on the genetic programming tool DataModeler. Real-time prediction of energy production of the wind farm Woolnorth in Tasmania, Australia was made based on weather and energy and for the creation of models and prediction, authors associated the wind farm with the Australian weather station ID091245, located at Cape Grim, Tasmania as its data is available for free for a running observation time window of 72 h. Thus, the prediction gave an accuracy of up to 80% R on the training range and up to 85.5% on the unseen test data.

Advantages:

Real-time data has been used which gives more accurate prediction.

Disadvantages:

The models proposed produce less accuracy than some machine learning like random forest algorithm

[5] SHORT-TERM WIND POWER PREDICTION BASED ON EXTREME LEARNING MACHINE WITH ERROR CORRECTION

In this paper, a combined approach based on Extreme Learning Machine (ELM) and an error correction model was proposed to predict wind power in the short-term time scale. Firstly, an ELM was utilized to forecast the short-term wind power. Then the ultra-short-term wind power forecasting was acquired based on processing the short-term forecasting error by persistence method. For short-term forecasting, the ELM did not perform well. The overall NRMSE (Normalized Root Mean Square Error) of forecasting results for 66 days is 21.09 %. For the ultra-short-term forecasting after error correction, most of forecasting errors lie in the interval of $[-10 \text{ MW}, 10 \text{ MW}]$. The error distribution was concentrated and almost unbiased. The overall NRMSE is 5.76 %.

The proposed model was then verified using the measured data in a wind farm located in the northern China for a period of about 15 months from 24 February 2014 to 31 May 2015. The 41072 non-consecutive data points before 02 March 2015 are used for training the ELM models whereas the consecutive time series of 66 days from 02 March 2015 to 31 May 2015 is used to verify the model's performances. The total installed capacity of the wind farm was 50 MW. The measured data are used for both training the ELM model and verifying the model. The time scale of collecting data is 15 min. The scatter of wind power versus wind speed of the wind farm was plotted in.

Advantages:

It is noted that the computation time was approximately proportional to the hidden nodes number. When the hidden nodes number was 3, the computation time is 1.5377 s, including the time for training model with 41072 data points and forecasting 6336 data points. That indicates a high computational efficiency for wind power forecasting, which can satisfy the practical needs.

Disadvantages:

The amount of data used to train the model was just a year's data and as climate varies from year to year, one year's data is not enough to get higher accuracy when this model is implementation and used for a number of years. Extreme Learning Machine (ELM) is not the best algorithm for using in large windfarm datasets though error correction models are used with ELM.

[6] FORECASTING WIND POWER GENERATION USING ARTIFICIAL NEURAL NETWORK:"PAWAN DANAWI"- A CASE STUDY FROM SRI LANKA

This paper develops artificial neural network (ANN) models to forecast wind power generation in "Pawan Danawi", a functioning wind farm in Sri Lanka. Monthly average power generation data (MW) from January 2015 to December 2019 (5 years, 60 data sets) were obtained from the wind farm authorities. Wind speed, wind direction, and ambient temperature of the area were used as the independent variable matrices of the developed ANN models, while the generated wind power was used as the dependent variable. ANN models were used because ANN reported suitable to forecast the wing power generation accurately outperforming other techniques such as support vector machine (SVM), neural network (NN), random forest (RF), and k-nearest neighbor (k-NN) and fuzzy logic techniques.

Then the models were tested with three training algorithms, namely, LevenbergMarquardt (LM), Scaled Conjugate Gradient (SCG), and Bayesian Regularization (BR) training algorithms. In addition, the model was calibrated for five validation percentages (5% to 25% in 5% intervals) under each algorithm to identify the best training algorithm with the most suitable training and validation percentages. Mean squared error (MSE), coefficient of correlation (R), root mean squared error ratio (RSR), Nash number, and BIAS were used to evaluate the performance of the developed ANN models. Results revealed that all three training algorithms produce acceptable predictions for the power generation in the Pawan Danawi wind farm with $R > 0.91$, $MSE < 0.22$, and $BIAS < 1$. LM and SCG algorithms have reached the optimum result in less time. Among them, the LM training algorithm at 70% of training and 5% of validation percentages produces the best forecasting results and also gives the lowest MSE at epoch 2.

Advantages:

- When considering the computational efficiency of the three algorithms, LM and SCG produce minimum MSE at relatively lesser epochs.
- Can work efficiently for similar environmental and climatic conditions (off shore) to identify the wind power potential of the area.
- According to the results, the error is negligible in most of the months and numerically less than 0.9 MW demonstrating the accuracy of the LM-based

ANN model. As per the calculations, it exhibits an RSR of 0.524 and a Nash number of -0.723 .

Disadvantages:

- Unlike the LM and SCG training algorithms, BR training algorithms show the best performance at 15% of validation percentage at epoch 922.
- However, the minimum MSE values of the BR algorithm are smaller than those of the other two algorithms only at a much higher epoch.
- In addition, there is a strong correlation between input parameters and power output at all five validation percentages in the BR algorithm. This can be advantageous in some cases but in some might not.
- The model works best for off shore wind farms but it may not work best for onshore wind farms as the data values, climate obtained varies with on shore values.

[7] DEEP LEARNING-BASED PREDICTION OF WIND POWER FOR MULTI-TURBINES IN A WIND FARM

In this paper, a deep learning approach was proposed for the power prediction of multiple wind turbines as the difficulty of conducting the temporal-spatial sequence prediction, such as “wind turbine power prediction” was to simultaneously extract the time dependence and spatial features hidden in the data. Therefore starting from the time series, after data preprocessing, the LSTM-CNN joint prediction model was proposed, which exploits a two-stage modeling strategy. Then, a deep neural network combines spatio temporal correlation to simultaneously predict the power of multiple wind turbines. Specifically, the network was a joint model composed of Long Short-Term Memory Network (LSTM) and Convolutional Neural Network (CNN). Herein, the LSTM captured the temporal dependence of the historical power sequence, while the CNN extracted the spatial features among the data, thereby achieving the power prediction for multiple wind turbines at different locations. Rectified Linear Unit (ReLU) is selected as the activation function of the convolutional layer which can improve the performance of CNN. The proposed approach was validated by using the wind power data from an offshore wind farm in China, and the results in comparison with other approaches shows the high prediction preciseness achieved by the proposed approach.

Advantages:

- The comparison results showed that the proposed joint method of CNN-LSTM has more excellent performance on predicting the wind power of multiple wind turbines within a wind farm with the regular layout than the existing prediction methods, such as LSTM, CNN, and SVM.
- On this basis, it can be used to perform accurate power scheduling.
- As CNN can extract the spatial features among the data, the power prediction for multiple wind turbines at different locations can be achieved.

Disadvantages:

This paper just uses SCADA data of 34 wind turbines at Guishan offshore wind farm in China which was collected within 1 week of November 2019. This data is not enough to predict energy output from wind turbines for a longer run.

[8] A PREDICTION MODEL FOR WIND FARM POWER GENERATION BASED ON FUZZY MODELING

In this paper, based on the historical data of a wind farm, the application of a fuzzy model for wind power prediction is carried out. It has been found that wind power can be successfully predicted using a fuzzy model.

Advantages:

- The prediction by use of the fuzzy model has the smallest error when the period is 0.5 hour.
- The model not only maintains good prediction accuracy but also provides an interpretable model structure which contains several rules, from which it may also reveal a useful qualitative description of the prediction system.

Disadvantages:

This model must be improved by considering effects of other factors like wind direction, humidity, etc as parameters.

2.1 EXISTING PROBLEM:

In almost all the existing solutions either some or all of the following parameters are missing:

1. Day-to-day weather and location are not taken as parameters for prediction.
2. Does not include problems due to line breakdown, power flow and power quality issues as parameters and generation-load prediction matching is not shown.
3. Height of the Wind Mill has not been taken as a parameter.
4. Dataset used is collected only for a short period of time.

2.2 REFERENCES

1. Haroon Rashid, Waqar Haider, Canras Batunlu Forecasting of Wind Turbine Output Power Using Machine learning IEEE 30 September 2020
2. Tarek H. M. El-Fouly and Ehab F. El-Saadany Grey Predictors for Hourly Wind Speed and Power Forecasting Springer 01 January 2012
3. Andy Clifton, Levi Kilcher, Julie Kay Lundquist, Paul Aaron Fleming Using machine learning to predict wind turbine power output IOP June 2013
4. Ekaterina Vladislavleva, Tobias Friedrich, Frank Neumann, Markus Wagner Predicting the energy output of wind farms based on weather data: Important variables and their correlation Elsevier February 2013
5. Zhi Li, Lin Ye, Yongning Zhao, Xuri Song, Jingzhu Teng & Jingxin Jin Short-term wind power prediction based on extreme learning machine with error correction Springer 2016
6. Amila T. Peiris, Jeevani Jayasinghe, and Upaka Rathnayake Forecasting Wind Power Generation Using Artificial Neural Network: "Pawan Danawi"- A Case Study From Sri Lanka Hindawi 24 Mar 2021
7. Xiaojiao Chen, Xiuqing Zhang, Mi Dong and Liansheng Huang Deep Learning-Based Prediction of Wind Power for Multi-turbines in a Wind Farm Frontiers in 29 July 2021
8. BoZhu, Min-youChen, NealWade and LiRan A prediction model for wind farm power generation based on fuzzy modelling Elsevier 6 April 2012

2.3 PROBLEM STATEMENT DEFINITION

Any Windfarm owner needs a Wind energy output prediction software that not only gives a prediction value closest to the actual output value but also gives notifications in case of unexpected weather changes to meet their customer demands with more satisfaction so that their customers can purchase power from other sources in case of less output or zero power generation due to any failures.

Ref: [Problem Statement, Brainstorming and Prioritization of ideas.pdf](#)



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Wind energy producer	Make prediction of energy output of wind turbine based on weather conditions	Current solutions lack in featuring prediction with varying weather conditions	The Dataset collected in existing solutions is for short time and it does not contain weather parameters	Frustrated and dissatisfied
PS-2	Wind energy producer	Get a prediction software that not only gives a prediction value closest to the actual output but also gives notifications in case of unexpected weather changes to meet customer demands so that my customers can purchase power from other sources when expected power can't be generated.	Current solutions do not take unexpected weather changes into account. And it also does not have a feature to notify users.	The machine learning model is not being fed with weather conditions. Notification is not built within the software in current solutions	Dissatisfied And sad

CHAPTER 3

IDEATION AND PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

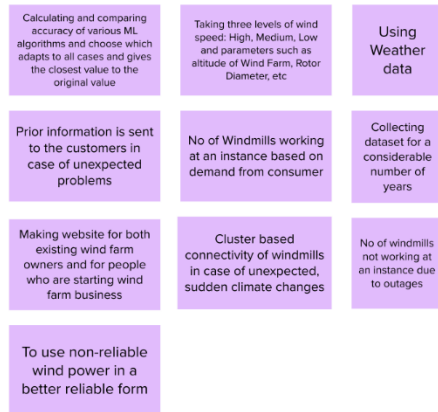
Wind power generation differs from conventional thermal generation due to the stochastic nature of wind. Thus wind power forecasting plays a key role in dealing with the challenges of balancing supply and demand in any electricity system, given the uncertainty associated with the wind farm power output. Accurate wind power forecasting reduces the need for additional balancing energy and reserve power to integrate wind power. For a wind farm that converts wind energy into electrical power, a real-time prediction system of the output power is significant. To develop this real-time prediction system, we need to understand end-user's perspective, feelings, thoughts and problems as adapting best approaches, finding and implementing solutions for these problems can create a huge change in prediction of output power.



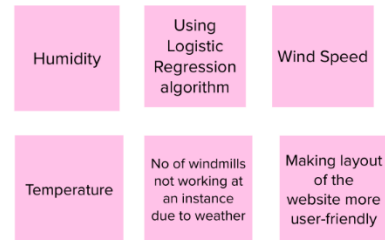
3.2 IDEATION AND BRAINSTORMING

Different ideas were generated for the project.

S.R. VIDHYAMBIKA (TEAM LEAD)



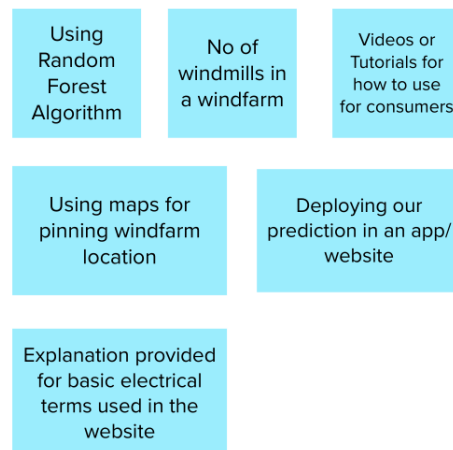
R.K. NAVINA



S. PRIYADHARSHINI



A. JEEVIKA

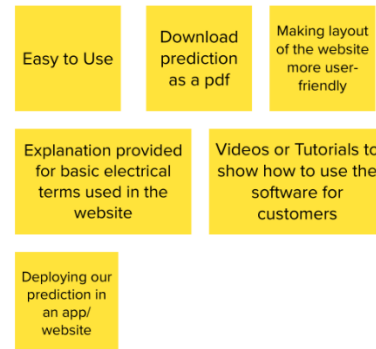


Then, these ideas are grouped into 4 categories as follows

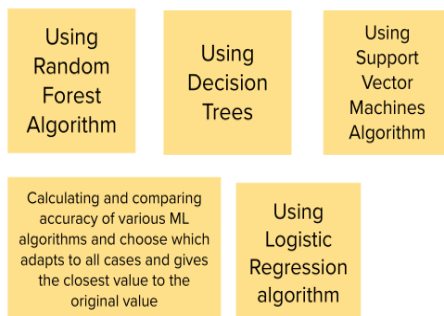
PARAMETERS TAKEN FOR PREDICTION



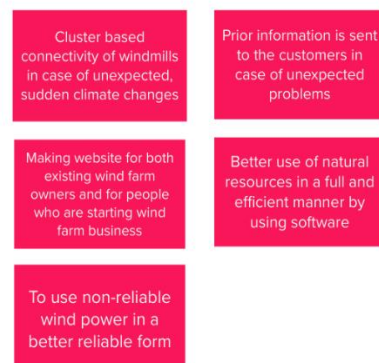
PREDICTION SOFTWARE



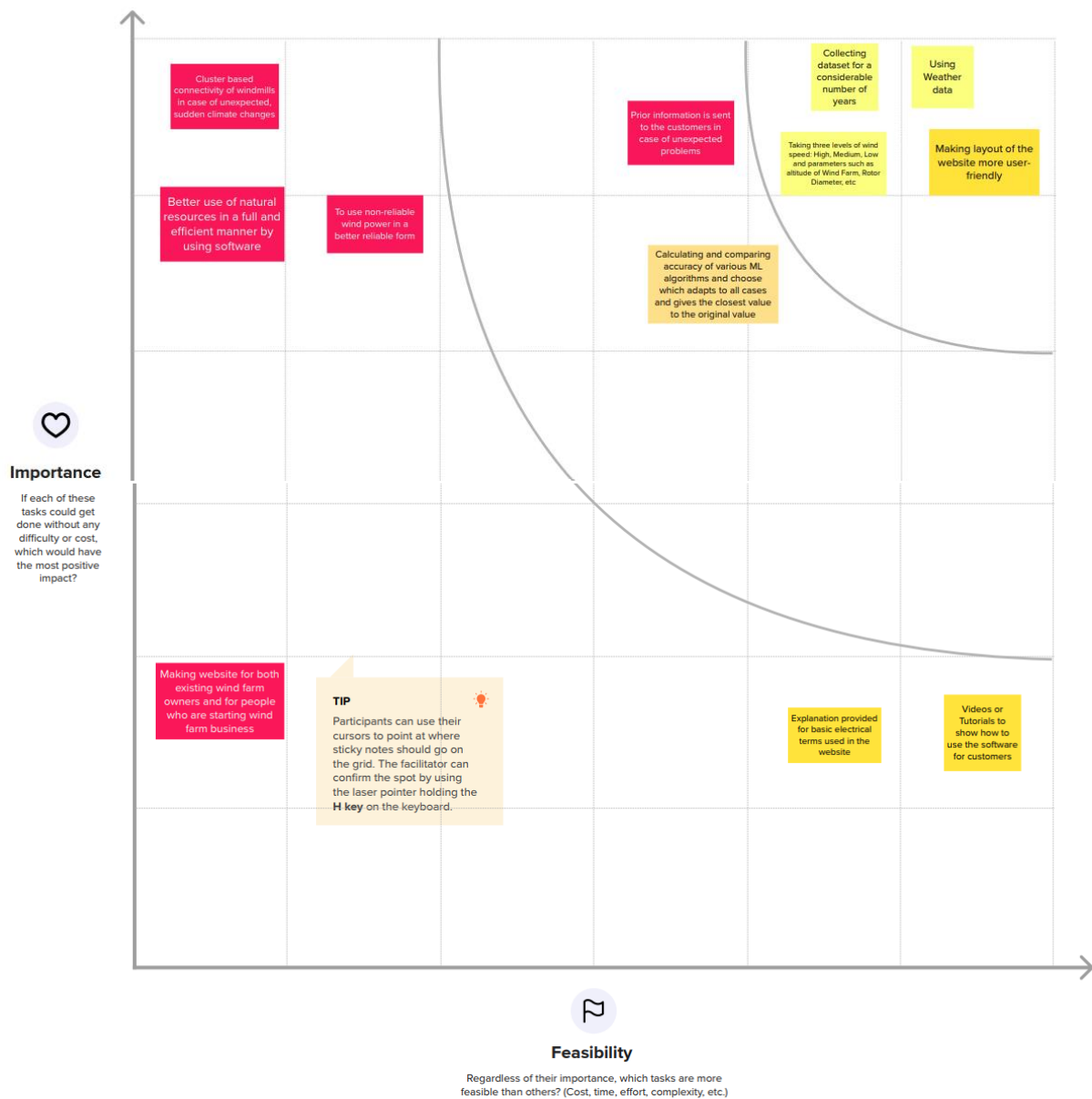
MACHINE LEARNING FOR PREDICTION



IMPROVING CUSTOMER SATISFACTION



Then these categorized ideas were displayed on different positions of graph (based on x-Feasibility y-Importance)







3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Any Windfarm owner needs a Wind energy output prediction software that not only gives a prediction value closest to the actual output value but also gives notifications in case of unexpected weather changes to meet their customer demands with more satisfaction so that their customers can purchase power from other sources in case of less output or zero power generation due to any failures.
2.	Idea / Solution description	With weather, Wind Mill altitude, Location, and Wind speed as parameters, different ML algorithms will be trained and tested. Then, the best-performing algorithm that gives the highest accuracy will be deployed as a web app. The Web app will be user-friendly, with an explanation of the electrical terms used. A notification system will be used to notify users priorly if the expected power cannot be generated due to unexpected weather changes like sudden rainfall and outages.
3.	Novelty / Uniqueness	Present wind energy output predicting solutions do not have any specific methods to predict the output energy based on the changing weather and wind conditions. These solutions also do not handle unexpected weather changes and outages. By implementing the proposed solution, Prediction of wind power output will be more accurate as dynamic weather, wind conditions are taken into account and as the prior notification system is introduced in this proposed solution, it will handle unexpected weather changes and outages.
4.	Social Impact / Customer Satisfaction	Meeting secondary consumers' demands as accurate prediction helps primary consumers (Wind Farm Owners, Wind power producers) to make satisfiable demands from secondary consumers and generate demanded power. So, this provides customer satisfaction to both primary and secondary consumers.

S.No.	Parameter	Description
5.	Business Model (Revenue Model)	Wind power forecast/prediction is critically required to increase the reliability of renewable energy generation. Accurate wind power generation prediction gives scope to the wind mill producers to sell maximum power to grid by effective scheduling so that all the power generated can be evacuated by the power utility.
6.	Scalability of the Solution	As data grows upon collecting and storing predicted values, actual output, wind speed, and weather in the cloud storage, the prediction software will maintain and utilize all these data and develop more accurate predictions in the upcoming years.

3.4 PROBLEM SOLUTION FIT

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) <small>Who is your customer? I.e. working parents of 0-5 y.o. kids</small> CS Wind Farm Owners & Wind Power Producers	6. CUSTOMER CONSTRAINTS <small>What constraints prevent your customers from taking action or limit their choices of solutions? I.e. spending power, budget, no cash, network connection, available devices.</small> CC Lack of Expertise	5. AVAILABLE SOLUTIONS <small>Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? I.e. pen and paper is an alternative to digital notetaking</small> AS Using SCADA historical data and Machine learning algorithms for prediction Using unsupervised learning and calibrating errors for prediction Generating a mathematical curve along with prediction	Explore AS, differentiate
	2. JOBS-TO-BE-DONE / PROBLEMS <small>Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.</small> J&P Problems such as Change of weather, Change of seasons, Natural Calamity, Cyclone, Failure, Outages, etc which need to be addressed to the consumers for which prediction of maximum wind power generation is required	9. PROBLEM ROOT CAUSE <small>What is the real reason that this problem exists? What is the back story behind the need to do this job? I.e. customers have to do it because of the change in regulations.</small> RC The main limitation in wind energy prediction is due to varying weather conditions, inconsistent nature of the wind which causes inconsistent power output High investment but less and delayed returns.	7. BEHAVIOUR <small>What does your customer do to address the problem and get the job done? [E] Directly related: find the right solar panel installer, calculate usage and benefits; [I] Indirectly associated: customers spend free time on volunteering work (I.e. Greenpeace)</small> BE Being a wind power producer, he is able to give accurate wind power output to his consumers and this satisfies both primary and secondary consumers. He is able to tap the maximum wind power potential based on the prediction. This gives him more financial benefits.	
Focus on J&P, tap into BE, understand RC				Focus on J&P, tap into BE, understand RC

Identify strong TR & EM	<p>3. TRIGGERS </p> <p>What triggers customers to act? I.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.</p> <p>If the customer finds it as an efficient solution, it will be automatically recommended by him/her to other wind power producers for better operation of wind power plants. The power utility(like TNEB/TANGEDCO) will recommend this solution to other wind power producers.</p>	<p>10. YOUR SOLUTION </p> <p>If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.</p>	<p>8.CHANNELS of BEHAVIOUR </p>
	<p>4. EMOTIONS: BEFORE / AFTER </p> <p>How do customers feel when they face a problem or a job and afterwards? I.e. lost, insecure > confident, in control - use it in your communication strategy & design.</p> <p>Before: Perplexed with improper energy flow and gets frustrated</p> <p>After: Happy with the efficient technique for maximum generation and evacuation</p>	<p>With weather, Wind Mill altitude, Location, Wind speed as parameters, different ML algorithms will be trained and tested. Then, the best performing algorithm which gives the highest accuracy will be deployed as a web app. The Web app will be user-friendly, with an explanation of electrical terms used. A notification system will be used to notify users priorly if the expected power can't be generated due to unexpected weather changes like sudden rainfall or outages.</p>	<p>8.1 ONLINE What kind of actions do customers take online? Extract online channels from #7</p> <p>8.2 OFFLINE What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.</p> <p>Offline: Ask field experts, Monitoring and maintaining wind farms, Having discussion with other Wind farm producers and TNEB in this regard Online: Searching in google, Contacting experts online through professional platforms, Update Offline production data</p>

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Gmail
FR-2	User Confirmation	Confirmation via OTP
FR-3	User Login	Login by entering credentials
FR-4	Reset Password	Change the password by using forgot/reset password option
FR-5	Information on wind energy prediction	Display information about the software (wind energy prediction with electrical terms used - explained)
FR-6	Account Settings & Edit Profile	Provide edit options for making changes in profile and account
FR-7	Update Actual Output produced	Prompt the user to update actual output if he/she is already an existing user
FR-8	Get & validate the location	Prompt the user to point to the location of the windmill and validate the location
FR-9	Get the Height of the Windmill	Prompt the user to enter the height of the windmill
FR-10	Displaying Prediction	Make predictions based on inputs from the user, weather, and wind speed, and display the predicted value to the user.
FR-11	Download Prediction Results	Download as image Download as PDF
FR-12	View Previous prediction results	View previous predictions and actual wind energy output values
FR-13	Prior-Notification	Notify secondary consumers of Wind energy producers if expected power cannot be generated after viewing predicted value, climate change, and outages.
FR-14	Admin Login	Admin login by entering credentials
FR-15	Add Features	Add features based on improvements in technologies
FR-16	Content Optimization	Add/Remove/Modify contents
FR-17	Accounts Management	Edit permission settings for accounts Managing user accounts
FR-18	Customer Data Management	1)Store and manage predicted values, live data, actual power output 2)Make and Store comparison table/curve of Actual Output and Predicted Output
FR-19	Customer Care Support	Manage customer queries

4.2 NON-FUNCTIONAL REQUIREMENTS

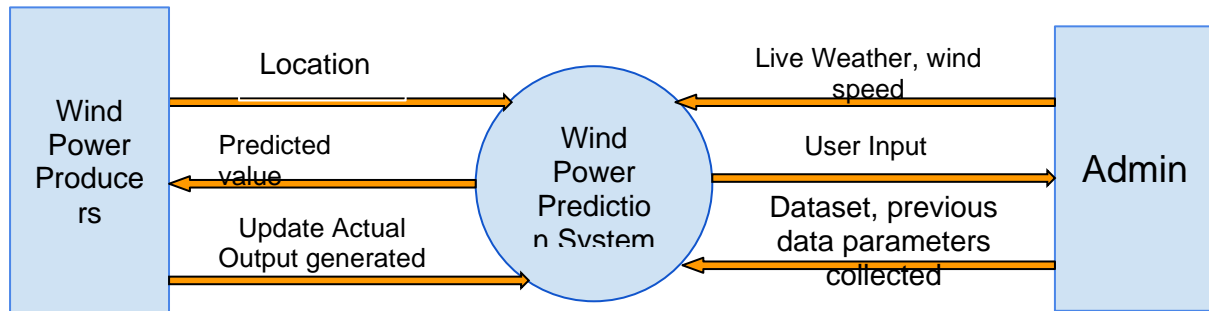
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	1) User-Friendly 2) Customized prediction based on each wind energy producer's wind farm parameters. 3) Electrical, Technical Terms used are explained within the software
NFR-2	Security	1) Secured cloud data 2) Allow users to use only passwords which satisfy the constraints 3) Provide 2-Factor Authentication
NFR-3	Reliability	1) Handle the less output-predicted situations due to unexpected weather changes and outages by sending notifications. 2) Maintain software, network traffic, website 3) Resolving technical issues occurring in short-time
NFR-4	Performance	Gives the most accurate prediction in short-time
NFR-5	Availability	1) Always available online 2) Supports most of the device configurations
NFR-6	Scalability	1) As data grows upon collection, storing, and maintenance of predicted values, actual output, wind speed, and weather in the cloud storage, utilizing all these data gives more accurate predictions in the upcoming years. 2) Modifying features often as data grows and technology improves to avoid large changes over a period

CHAPTER 5

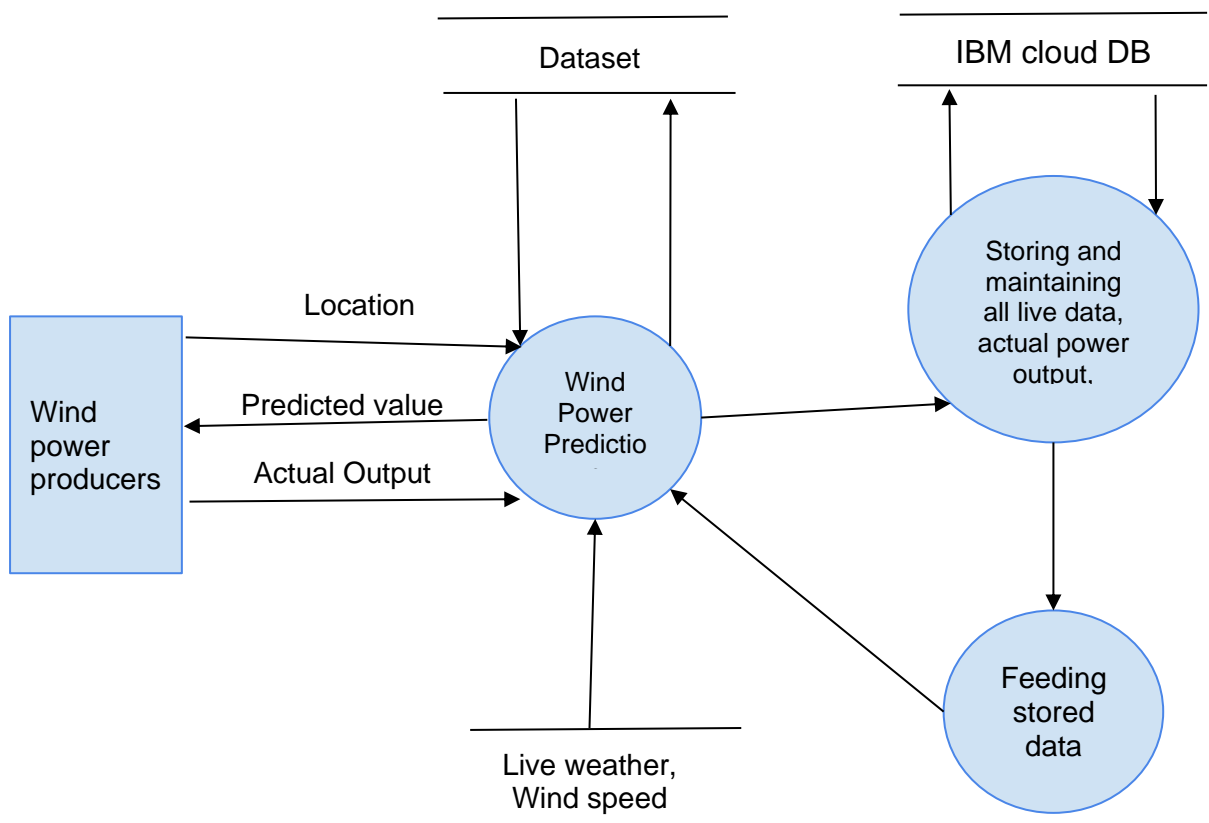
PROJECT DESIGN

5.1 DATA FLOW DIAGRAM:

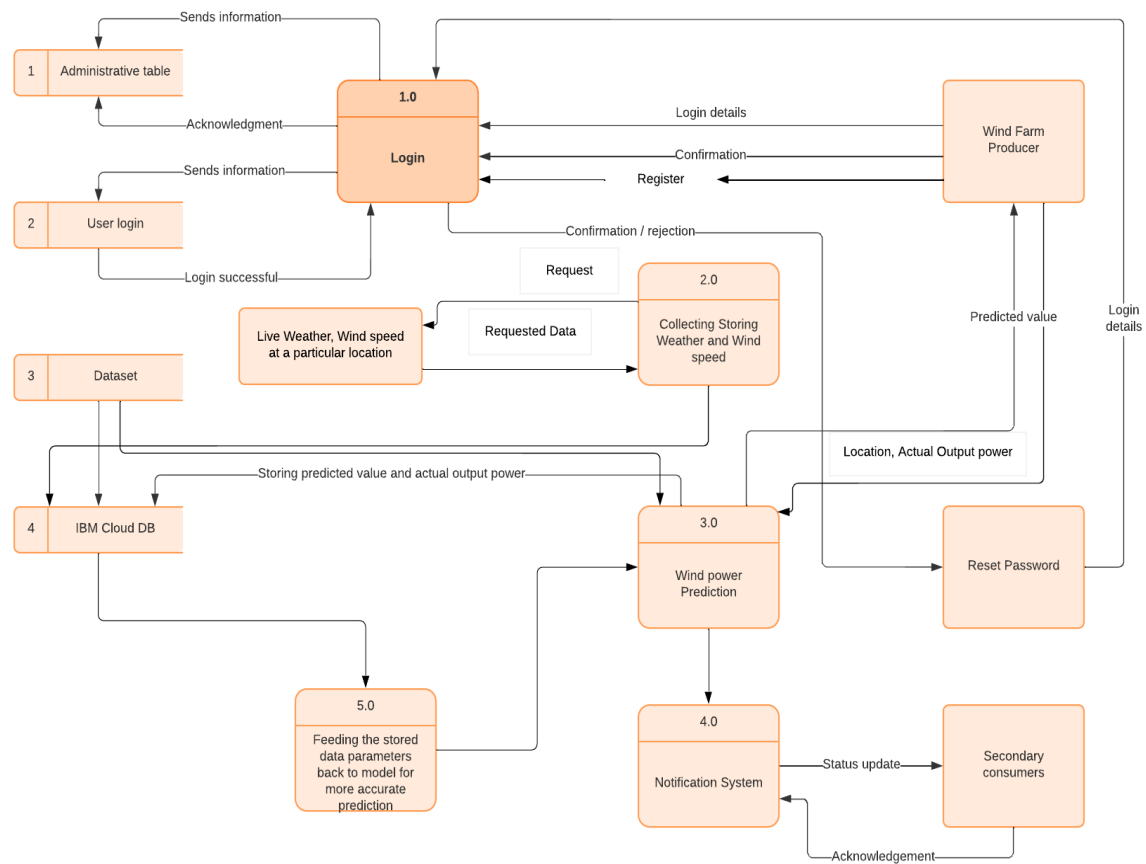
LEVEL 0



LEVEL 1



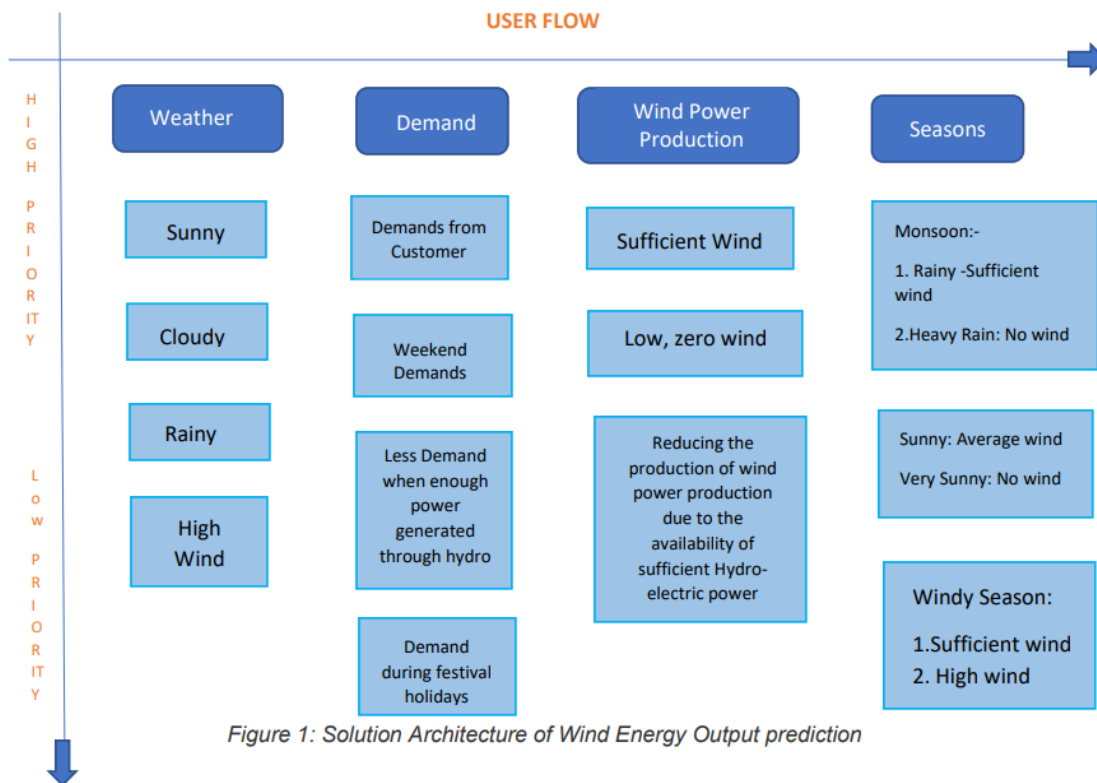
FINAL DFD DIAGRAM (BASED ON GANE-SARSON MODEL)



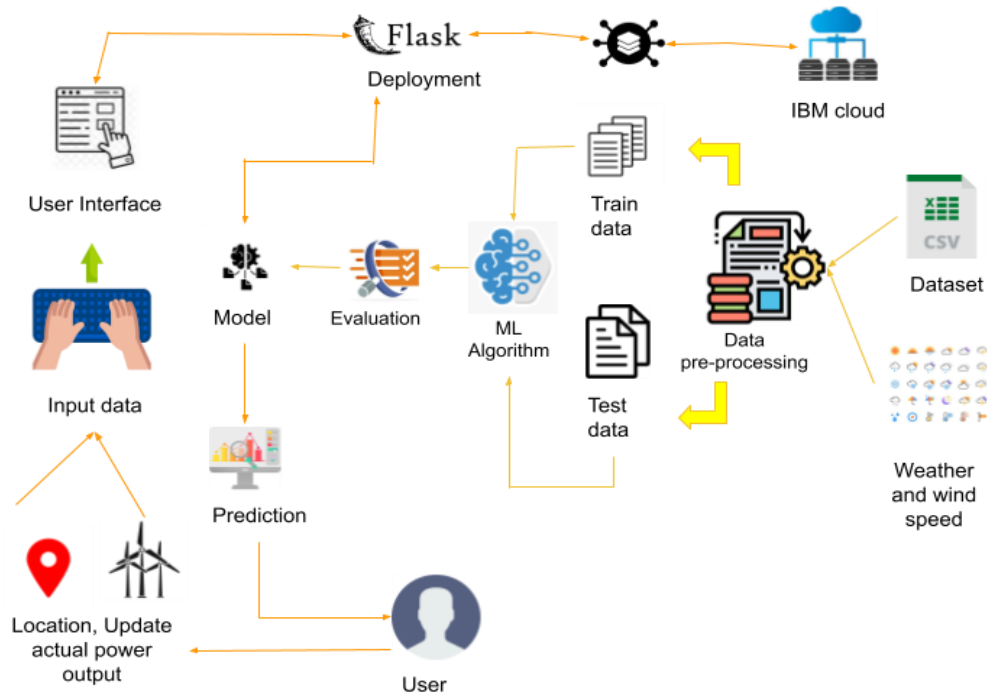
Note:

1. No wind, zero situations mean there isn't sufficient wind to produce electricity.
2. High wind means the wind speed is very high and running windmills might damage them.

5.2 TECHNOLOGY AND SOLUTION ARCHITECTURE



SOLUTION ARCHITECTURE



TECHNOLOGY ARCHITECTURE

5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Wind power producer)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can access my account / dashboard	High	Sprint-1
		USN-4	As a user, I can use Forgot Password option to reset my password if I forgot my current password	I can reset password through mail received for password reset	Medium	Sprint-2
	Dashboard	USN-5	As a user, I can access the Dashboard to view my profile	I can view my profile	Low	Sprint-3
		USN-6	As a user, I can access the Dashboard to give my location, actual output power for prediction	I can get customised prediction value for my wind farm	High	Sprint-1
	Profile	USN-7	As a user, I can edit my profile from time to time	I can update my profile	Low	Sprint-3
	Notification	USN-8	Sending notifications to secondary consumers if predicted output will not be able to generate due to unexpected weather changes and outages	I can send notifications thus my customers are satisfied as notifications are sent priorly	High	Sprint-1
Administrator	IBM cloud	USN-9	As an administrator, I can store actual output power, predicted value, weather and wind speed	I store and maintain these data	High	Sprint-1
	IBM cloud	USN-10	As an administrator, I can feed all the stored data as parameters back to the model for more accurate prediction	I can train the model with more data to get accurate prediction	High	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
	Login	USN-11	As an administrator, I can reset password for customers	I can help customers to get back their access to account	Medium	Sprint-2

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, and password, and confirming my password.	2	High	VIDHYAMBIKA S.R
Sprint-1		USN-2	As a user, I will receive a confirmation email once I have registered for the application	1	High	VIDHYAMBIKA S.R
Sprint-1	Login	USN-3	As a user, I can log into the application by entering my email & password	2	High	VIDHYAMBIKA S.R
Sprint-2		USN-4	As a user, I can use Forgot Password option to reset my password if I forgot my current password	2	Medium	VIDHYAMBIKA S.R
Sprint-3	Dashboard	USN-5	As a user, I can access the Dashboard to view my profile	1	Low	VIDHYAMBIKA S.R
Sprint-1		USN-6	As a user, I can access the Dashboard to give my location, actual output power for prediction	2	High	VIDHYAMBIKA S.R
Sprint-1		USN-7	As a user, I can enter the height of the Wind Mill	2	High	VIDHYAMBIKA S.R
Sprint-1		USN-8	As a user, I can make a prediction	2	High	VIDHYAMBIKA S.R
Sprint-3	Profile	USN-9	As a user, I can edit my profile from time to time	1	Low	VIDHYAMBIKA S.R
Sprint-1	View Prediction	USN-11	As a user, I can view the prediction made	2	High	VIDHYAMBIKA S.R
Sprint-3	Download Prediction	USN-12	As a user, I can download the prediction either as pdf or png	1	Low	VIDHYAMBIKA S.R
Sprint-1	Notification	USN-13	As a user, I can send notifications to secondary consumers if the predicted output will not be able to generate due to unexpected weather changes and outages	2	High	VIDHYAMBIKA S.R
Sprint-4	View previous values	USN-14	As a user, I can view previously predicted values, actual output values, wind speed at that time, and weather.	2	Low	VIDHYAMBIKA S.R
Sprint-1	IBM Cloud	USN-15	As an administrator, I can store actual output power,	2	High	VIDHYAMBIKA S.R

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
			predicted value, weather, and wind speed			
Sprint-1		USN-16	As an administrator, I can feed all the stored data as parameters back to the model for more accurate prediction	2	High	VIDHYAMBIKA S.R
Sprint-2	Login	USN-17	As an administrator, I can reset passwords for customers	2	Medium	VIDHYAMBIKA S.R
Sprint-4		USN-18	As an administrator, I can reset password using Forgot password option	1	Low	VIDHYAMBIKA S.R
Sprint-4	Content Optimization	USN-19	As an administrator, I can Add/Remove/Modify contents as required	1	Low	VIDHYAMBIKA S.R
Sprint-4	Accounts Management	USN-20	As an administrator, I can edit permission settings for accounts and manage user accounts	1	Low	VIDHYAMBIKA S.R
Sprint-2	Customer Data Management	USN-21	As an administrator, I can store and manage predicted values, live data, actual power output	2	Medium	VIDHYAMBIKA S.R
Sprint-3		USN-22	As an administrator, I can make and Store comparison table / curve of Actual Output and Predicted Output for users	2	Low	VIDHYAMBIKA S.R
Sprint-3	Customer Care Support	USN-23	Manage customer queries	2	Low	VIDHYAMBIKA S.R

6.2 SPRINT DELIVERY SCHEDULE

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	19	6 Days	24 Oct 2022	29 Oct 2022	19	6 Nov 2022
Sprint-2	8	6 Days	31 Oct 2022	05 Nov 2022	8	9 Nov 2022
Sprint-3	6	6 Days	07 Nov 2022	12 Nov 2022	6	12 Nov 2022
Sprint-4	4	6 Days	14 Nov 2022	19 Nov 2022	4	19 Nov 2022

Velocity:

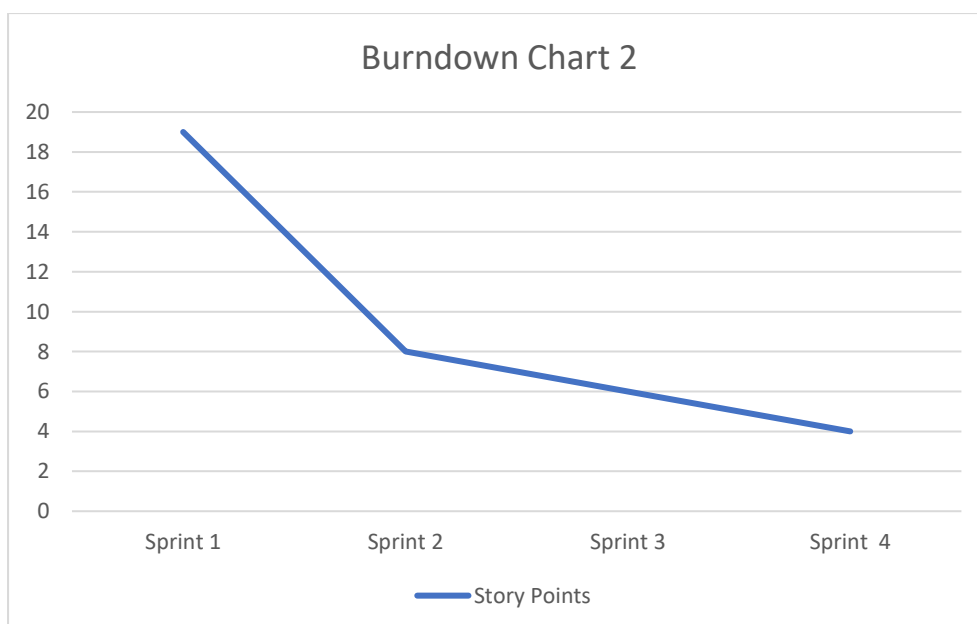
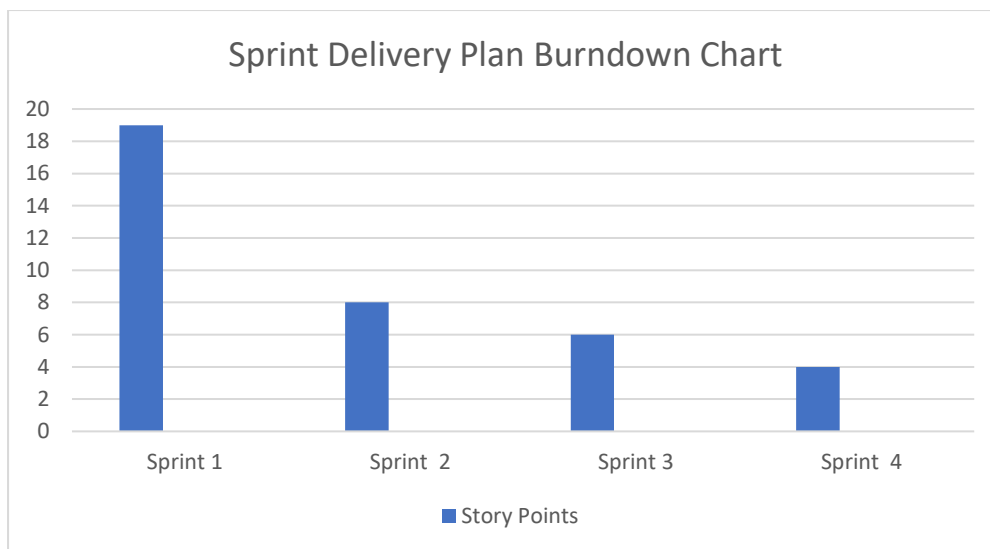
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

Jira Website: <https://pnt2022tmid31390.atlassian.net/>

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



6.3 REPORTS FROM JIRA

SPRINT 1

Projects / Wind Turbine Energy Output Nalaiya Thiran based on Weather / WTEONTBOW board

WTEONTBO Sprint 1

0 days remaining Complete sprint

Search this board VS Only My Issues Recently Updated Insights

TO DO	IN PROGRESS	DONE
		<p>Registration 3 WTEONTBOW-1 VS Assignee: Vithiyambika SR</p> <p>Login 2 WTEONTBOW-2 VS</p> <p>Prediction 12 WTEONTBOW-3 VS</p> <p>Notification 2 WTEONTBOW-4 VS</p>

SPRINT 2

Jira Software Your work Projects Filters Dashboards People Apps Create

Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Wind Turbine Energy Output Nalaiya Thiran based on Weather / WTEONTBOW board

WTEONTBO Sprint 2

0 days remaining Complete sprint

Search this board VS Only My Issues Recently Updated Insights

TO DO	IN PROGRESS	DONE
		<p>Deployment in IBM WTEONTBOW-9 VS Assignee: Vithiyambika SR</p> <p>Admin Login WTEONTBOW-12 VS</p> <p>Storing user credentials - database WTEONTBOW-13 VS</p>

Quickstart

SPRINT 3

Jira Software Your work Projects Filters Dashboards People Apps Create

Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Wind Turbine Energy Output Nalaiya Thiran based on Weather / WTEONTBOW board

WTEONTBO Sprint 3

0 days remaining Complete sprint

Search this board VS Only My Issues Recently Updated Insights

TO DO	IN PROGRESS	DONE
		<p>Dashboard WTEONTBOW-14 VS Assignee: Vithiyambika SR</p> <p>Integrating the weather api WTEONTBOW-16 VS</p> <p>Styling the website WTEONTBOW-15 VS</p>

Quickstart

SPRINT 4

Jira Software Your work Projects Filters Dashboards People Apps Create

Does your team need more from Jira? Get a free trial of our Standard plan.

Projects / Wind Turbine Energy Output Nalaya Thiran based on Weather / WTEONTBOW board

WTEONTBO Sprint 4

0 days remaining Complete sprint

Search this board VS Only My Issues Recently Updated Insights

TO DO

IN PROGRESS

Logout WTEONTBOW-17 VS Assignee: Vidhyambika SR

Navigate to previous website WTEONTBOW-18 VS

Customer data management and customer support channels for users WTEONTBOW-20 VS

Content Optimization by admin WTEONTBOW-19 VS

Quickstart

ALL ISSUES:

Type	Key	Summary	Assignee	Reporter	P	Status	Resolution
	WTEONT...-20	Customer data management and customer support channels for users	VS Vidhyambika SR	VS Vidhyambika SR	▼	DONE ✓	Done
	WTEONT...-19	Content Optimization by admin	VS Vidhyambika SR	VS Vidhyambika SR	▼	DONE ✓	Done
	WTEONT...-18	Navigate to previous website	VS Vidhyambika SR	VS Vidhyambika SR	▼	DONE ✓	Done
	WTEONT...-17	Logout	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-16	Integrating the weather api	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-15	Styling the website	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-14	Dashboard	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-13	Storing user credentials - database	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-12	Admin Login	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-10	Storing user credentials - database	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
Type	Key	Summary	Assignee	Reporter	P	Status	Resolution
	WTEONT...-13	Storing user credentials - database	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-12	Admin Login	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-10	Storing user credentials - database	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-9	Deployment in IBM	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-8	Admin Login	VS Vidhyambika SR	VS Vidhyambika SR	=	DONE ✓	Done
	WTEONT...-4	Notification	VS Vidhyambika SR	VS Vidhyambika SR	⬆	DONE ✓	Done
	WTEONT...-3	Prediction	VS Vidhyambika SR	VS Vidhyambika SR	⬆	DONE ✓	Done
	WTEONT...-2	Login	VS Vidhyambika SR	VS Vidhyambika SR	⬆	DONE ✓	Done
	WTEONT...-1	Registration	VS Vidhyambika SR	VS Vidhyambika SR	⬆	DONE ✓	Done

CHAPTER 7

CODING AND SOLUTIONING

7.1 FEATURE 1:

Getting the weather of a particular location using Open Weather API

weathercheck.py (also added in app.py)

```
# import required modules
```

```
import requests, json
```

```
# Enter your API key here
```

```
api_key = "9dc73970faafde4beb008b5e93ca7ab1"
```

```
# base_url variable to store url
```

```
base_url = "http://api.openweathermap.org/data/2.5/weather?"
```

```
# Give city name
```

```
from app import cn
```

```
# complete_url variable to store
```

```
# complete url address
```

```
complete_url = base_url + "q=" + city_name + "&appid=" + api_key
```

```
# get method of requests module
```

```
# return response object
```

```
response = requests.get(complete_url)
```

```
# json method of response object
```

```
# convert json format data into
# python format data
x = response.json()

# Now x contains list of nested dictionaries
# Check the value of "cod" key is equal to
# "404", means city is found otherwise,
# city is not found
if x["cod"] != "404":

    # store the value of "main"
    # key in variable y
    y = x["main"]

    # store the value corresponding
    # to the "temp" key of y
    current_temperature = y["temp"]

    # store the value corresponding
    # to the "pressure" key of y
    current_pressure = y["pressure"]

    # store the value corresponding
    # to the "humidity" key of y
    current_humidity = y["humidity"]

    #store the value of "wind"
    # key in variable a
    a = x["wind"]
```



```

# store the value corresponding
# to the "wind speed" key of a
current_windspeed = a["speed"]
current_winddeg = a["deg"]

# store the value of "weather"
# key in variable z
z = x["weather"]

# store the value corresponding
# to the "description" key at
# the 0th index of z
weather_description = z[0]["description"]

# print following values
print(" Temperature (in kelvin unit) = " +
      str(current_temperature) +
      "\n atmospheric pressure (in hPa unit) = " +
      str(current_pressure) +
      "\n humidity (in percentage) = " +
      str(current_humidity) +
      "\n description = " +
      str(weather_description) +
      "\n wind speed (in meter per sec) = " +
      str(current_windspeed) +
      "\n wind direction (in degrees) = " +
      str(current_winddeg) )

else:
    print(" City Not Found ")

```

FEATURE 2:

Notification System to notify users how much power can't be generated based on predicted power value.

notificationssystem.py :

```
# importing required modules and libraries
import datetime # to read present date
import time # to suspend the execution for a specific time
import requests

from plyer import notification # to get notification on the computer
# initializing a variable with None (temporary)
# indicating that there is no data available currently
from app.py import predict

user = int(input("Enter 1 to send notification, enter 0 to not send a notification"))
demandpower = float(input("Enter demanded value"))
title = "Sorry, demanded power can't be generated for 2022 -11-19"
message = "Predicted value is {predict} but, Demanded value is {demandpower}"

# repeating the loop for multiple times
while(user ==1):
    notification.notify(
        # defining the title of the notification,
        title = title,
        # defining the message of the notification
        message = message,
        # creating icon for the notification
        # we have to download a icon of ico file format
        app_icon = "windmill.ico",
        # the notification stays for 30 seconds
        timeout = 30,
    )
    break
```

CHAPTER 8

TESTING

8.1 TEST CASES

REPORT: -

				Date	3-Nov-22								
				Team ID	PNT2022TMID31390								
				Project Name	Project - Predicting the energy ou								
				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_OO 1	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on My account button		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Signup popup displayed or not	Website generated upon deployment	Login/Signup popup should display	Working as expected	Pass				S.R. VIDHYAMBIKA
LoginPage_TC_OO 2	UI	Home Page	Verify the UI elements in Login/Signup popup		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Signup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link e.Last password? Recovery password link	Website generated upon deployment	Application should show below UI elements: a.email text box b.password text box c.Login button with orange colour d.New customer? Create account link e.Last password? Recovery password link	Working as expected	Pass				S.R. VIDHYAMBIKA
LoginPage_TC_OO 3	Functional	Home page	Verify user is able to log into application with Valid credentials		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: admin password: admin	User should navigate to user account homepage	Working as expected	Pass				S.R. VIDHYAMBIKA
Screenshot of the application showing the Login/Signup popup													
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_OO 4	Functional	Login page	Verify user is able to log into application with Invalid credentials		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter Invalid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: vidhyambika@gmail.com password: vidhyambika1234_	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				S.R. VIDHYAMBIKA
LoginPage_TC_OO 5	Functional	Login page	Verify user is able to log into application with Invalid credentials		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on login button	Username: nalaiyathiran31390@gmail.com password: Nalaiya456	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				S.R. VIDHYAMBIKA
LoginPage_TC_OO 6	Functional	Login page	Verify user is able to log into application with Invalid credentials		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter Invalid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on login button	Username: bmsrvidhya@ec.in password: abc	Application should show 'Incorrect email or password' validation message.	404 error	Fail				S.R. VIDHYAMBIKA

Ref: [Testcases Report Template PNT2022TMID31390.xlsx](#)

8.2 USER ACCEPTANCE TESTING

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of this project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	6	3	1	1	10
Duplicate	3	0	0	0	3
External	2	3	0	1	6
Fixed	11	4	1	1	17
Not Reproduced	0	0	1	0	1
Skipped	0	0	0	1	1
Won't Fix	0	2	0	0	2
Totals	22	12	3	4	40

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	7	0	0	7
Client Application	51	0	0	51
Security	5	0	0	5
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	15	0	0	15
Version Control	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

Model Performance Testing:

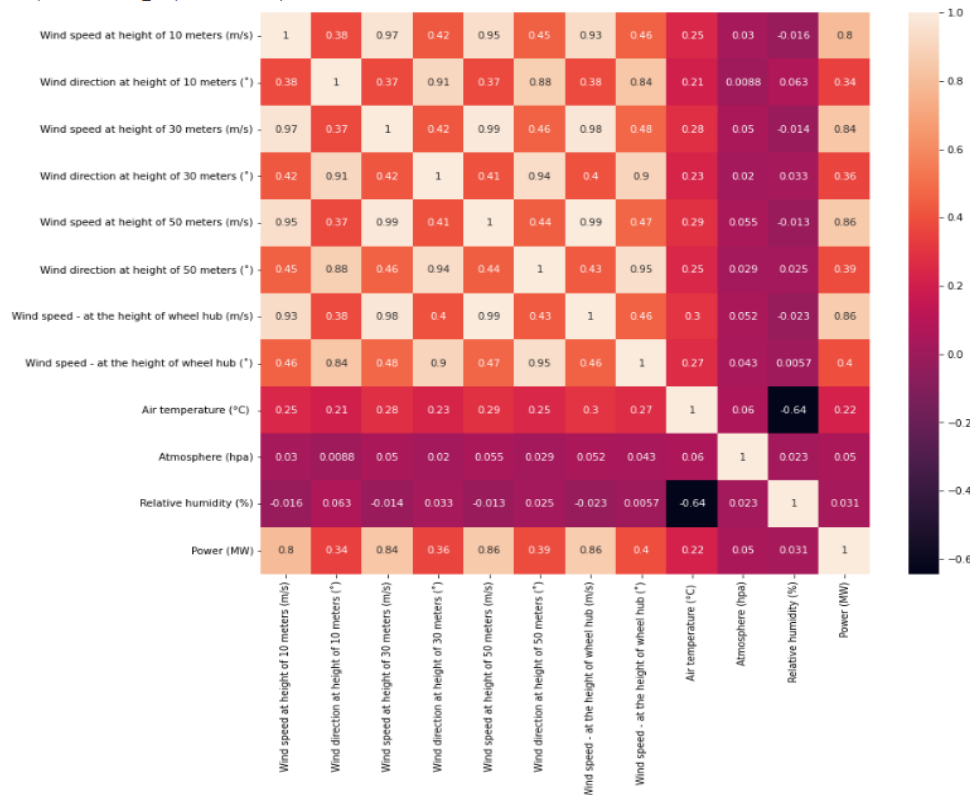
S.N o.	Parameter	Values	Screenshot																																				
1	Metrics	<p>Regression Model: MAE – 4.972 , MSE – 67.607, RMSE – 8.222, R2 score – 0.8849</p> <p>Random Forest Regressor</p> <p>Accuracy Score- 88.4%</p>	<p>Mean Absolute Error: 4.972486590131098 Mean Squared Error: 67.60709947932519 Root Mean Squared Error: 8.222353645965685</p> <pre>from sklearn.metrics import r2_score r2_score(y_test, y_pred) 0.8849999426996336</pre> <p>Ref: DEPLOYMENT-S_R_VIDHYAMBIKA.ipynb</p> <pre>Acc=pd.DataFrame({'Actual_y_value':y_test,'Predicted_y_value':y_pred}) Acc</pre> <table><thead><tr><th></th><th>Actual_y_value</th><th>Predicted_y_value</th></tr></thead><tbody><tr><td>50585</td><td>41.646313</td><td>33.260701</td></tr><tr><td>1610</td><td>26.752834</td><td>25.186579</td></tr><tr><td>68970</td><td>0.186879</td><td>0.328045</td></tr><tr><td>66444</td><td>1.792049</td><td>0.463834</td></tr><tr><td>38116</td><td>0.140692</td><td>0.483375</td></tr><tr><td>...</td><td>...</td><td>...</td></tr><tr><td>43785</td><td>7.865973</td><td>8.338973</td></tr><tr><td>34494</td><td>49.145634</td><td>51.113949</td></tr><tr><td>30848</td><td>1.079351</td><td>1.217201</td></tr><tr><td>33926</td><td>0.900288</td><td>0.494531</td></tr><tr><td>54458</td><td>84.859420</td><td>70.877957</td></tr></tbody></table> <p>21053 rows x 2 columns</p> <p>Ref:Model1_RandomForestRegressor_VIDHYAMBIKA.ipynb</p>		Actual_y_value	Predicted_y_value	50585	41.646313	33.260701	1610	26.752834	25.186579	68970	0.186879	0.328045	66444	1.792049	0.463834	38116	0.140692	0.483375	43785	7.865973	8.338973	34494	49.145634	51.113949	30848	1.079351	1.217201	33926	0.900288	0.494531	54458	84.859420	70.877957
	Actual_y_value	Predicted_y_value																																					
50585	41.646313	33.260701																																					
1610	26.752834	25.186579																																					
68970	0.186879	0.328045																																					
66444	1.792049	0.463834																																					
38116	0.140692	0.483375																																					
...																																					
43785	7.865973	8.338973																																					
34494	49.145634	51.113949																																					
30848	1.079351	1.217201																																					
33926	0.900288	0.494531																																					
54458	84.859420	70.877957																																					

2	Tune the Model	<p>Hyperparameter Tuning – Randomized search CV</p> <p>Validation Method -----></p>	<pre># Use the random grid to search for best hyperparameters # First create the base model to tune rf = RandomForestRegressor(n_estimators=10, random_state=0) # Random search of parameters, using 3 fold cross validation, # search across different conditions, and use all available cores rf_random = RandomizedSearchCV(estimator = rf, param_distributions = random_grid, n_iter = 3, cv = 3, verbose=2, random_state=4, n_jobs = -1) # Fit the random search model rf_random.fit(X_train, y_train) # Fit 3 folds for each of 3 candidates, totalling 9 fits for/local/lib/python3.7/dist-packages/joblib/external/loky/process_executor.py:782: UserWarning: A worker stopped while some jobs were given to it. This can be caused by a too short worker timeout or by a memory leak. "Timeout or by a memory leak.", UserWarning) RandomizedSearchCV(cv=3, estimator=RandomForestRegressor(n_estimators=10, random_state=0), n_iter=3, n_jobs=-1, param_distributions={'bootstrap': [True, False], 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None]}, 'max_features': ['auto', 'sqrt'], 'min_samples_leaf': [1, 2, 4], 'min_samples_split': [2, 5, 10], 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]}, random_state=4, verbose=2)</pre> <p>Ref: Model1 RandomForestRegressor VIDHYA MBIKA.ipynb</p> <pre>print('Best Score: %s' % result.best_score_)</pre> <p>Best Score: 0.8808203361658858</p> <p>Ref: DEPLOYMENT-S R VIDHYAMBIKA.ipynb</p>
---	----------------	--	--

CORRELATION HEATMAP:

```
plt.figure(figsize=[13,11])
sns.heatmap(df.corr(),annot=True)
```

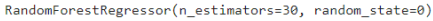
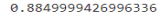
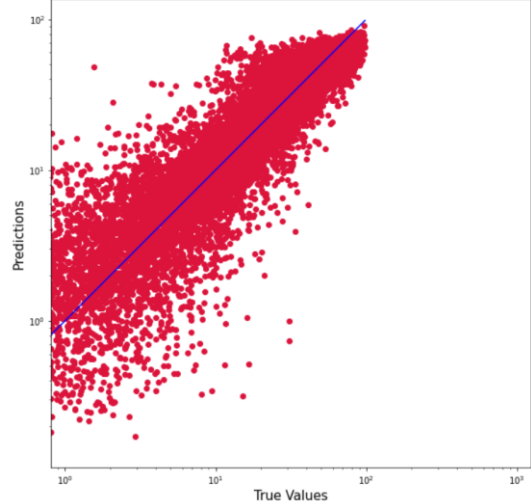
<matplotlib.axes._subplots.AxesSubplot at 0x7f202e4ddf90>



Ref: [DATA PRE PROCESSING BY VIDHYAMBIKA SR.pdf](#)

Different Models Performance Testing on same dataset:

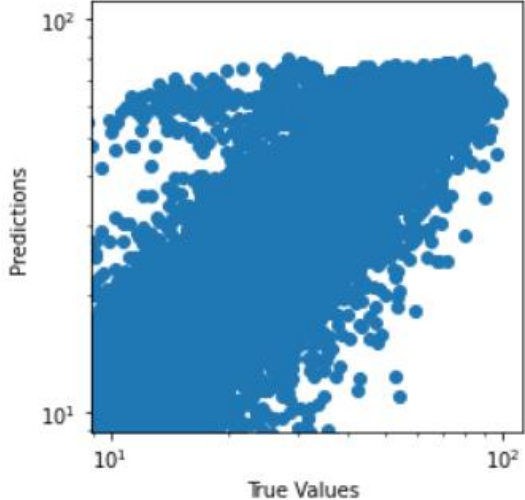
Random Forest Regressor:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Gives the highest accuracy. Random forest is a supervised learning algorithm that uses an ensemble learning method for classification and regression.	<pre>from sklearn.ensemble import RandomForestRegressor regressor = RandomForestRegressor(n_estimators=30, random_state=0) regressor.fit(X_train,y_train)</pre>  Ref:Model1_RandomForestRegressor_VIDH YAMBIKA.ipynb
2.	Accuracy	Training Accuracy – 100% Validation Accuracy – 88.4%	<pre>from sklearn.metrics import r2_score r2_score(y_test, y_pred)</pre>  Ref:Model1_RandomForestRegressor_VIDH YAMBIKA.ipynb
3.	Visualizing the model	X axis-True values Y axis-Predicted values	 Ref: VISUALISING_M1ipynb.ipynb

Random Forest Regressor but with two parameters eliminated:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Gives accuracy less than model 1. Random forest is a bagging technique and not a boosting technique.	<pre>from sklearn.ensemble import RandomForestRegressor regressor = RandomForestRegressor (n_estimators=20, random_state=0) regressor.fit(X_train,y_train)</pre> <p>RandomForestRegressor(n_estimators=20, random_state=0)</p>
2.	Accuracy	Training Accuracy – 100% Validation Accuracy – 87.7%	<pre>from sklearn.metrics import r2_score r2_score(y_test, y_pred)</pre> <p>0.8773928979826746</p> <p>Ref: Model2RandomForestRegelimatin g2par.ipynb</p>
3.	Visualizing the model	X axis-True values Y axis-Predicted values	 <p>Ref: VISUALISINGM2.ipynb</p>

Support Vector Regression (SVR):

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Gives accuracy less than model 2. SVR uses the same principle as SVM, but this SVR is applied for regression problems.	<pre> from sklearn.svm import SVR # Choose regression method and set hyperparameter svr_rbf=SVR(C=1.0, epsilon=0.2, kernel='rbf') # Training of the regression model svr_rbf.fit(X_train, y_train) y_pred = svr_rbf.predict(X_test) </pre>
2.	Accuracy	Training Accuracy – 100% Validation Accuracy – 84.6%	<pre> from sklearn.metrics import r2_score r2_score(y_test, y_pred) </pre> <p>0.8466893820599397</p> <p>Ref: Model3SVR.ipynb</p>
3.	Visualizing the model	X axis-True values Y axis-Predicted values	 <p>Ref: VISUALISING M3.ipynb</p>

Multi linear Regression:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Gives accuracy less than model 3. Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. Multiple regression is an extension of linear (OLS) regression that uses just one explanatory variable.	<pre>from sklearn.linear_model import LinearRegression lr=LinearRegression()</pre> <pre>lr.fit(X_train,y_train)</pre> <pre>LinearRegression()</pre> <pre>y_pred=lr.predict(X_test) y_pred</pre> <p>Ref: Model4multilinearregression.ipynb</p>
2.	Accuracy	Training Accuracy – 100% Validation Accuracy – 75%	<pre>print("Mean squared error =", round(sm.mean_squared_error(y_test, y_pred), 2)) print("Median absolute error =", round(sm.median_absolute_error(y_test, y_pred), 2)) print("Explain variance score =", round(sm.explained_variance_score(y_test, y_pred), 2)) print("R2 score =", round(sm.r2_score(y_test, y_pred), 2))</pre> <pre>Mean squared error = 149.42 Median absolute error = 6.21 Explain variance score = 0.75 R2 score = 0.75</pre>
3.	Visualizing the model	X axis-True values Y axis-Predicted values	<p>y_test vs y_pred (Test set)</p>  <p>Ref: VISUALISING M4.ipynb</p>

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

1. Better demand - generation matching and fulfilling the load demand since, this prediction gives better load management to grid
2. Maximum wind evacuation by the power utility and effective scheduling.
3. No loss of generation and maximum revenue realization for wind power producers.
4. Day, week and month ahead schedules based on weather with automatic updates based on weather makes the unreliable wind power a more reliable power.

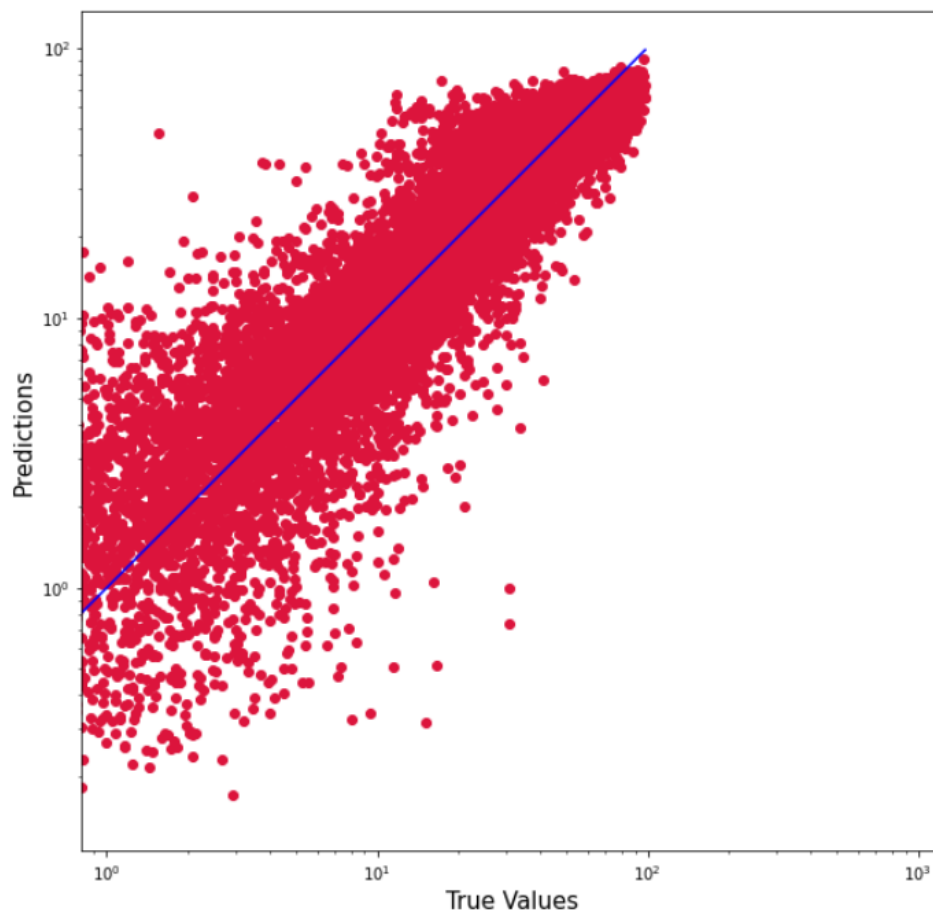
DISADVANTAGES

1. Could not meet demanded power in unexpected weather conditions
2. Surplus power generated cannot be stored.
3. Generated power transmission and distribution to the demanded areas is difficult and costly

CHAPTER 11

CONCLUSION

This project focused on predicting the energy output value closest to the actual value with weather parameters like Temperature, humidity & pressure and wind parameters like Wind speed & wind direction using Random Forest Regressor. The accuracy obtained is 88.4% and to handle the unexpected weather conditions, a notification system has been introduced for customers to notify their customers if demand power cannot be generated by seeing the predicted power.



CHAPTER 12

FUTURE WORK

This project can be extended by introducing following features:

- a. Cluster based management/connectivity of wind mills for less or no wind conditions. Here, less or no wind means power can't be generated in windmills with that wind speed or negative power (Wind power generated is less than the consumption power of wind mills running to generate that power)
- b. Better power management in case of micro-grid with wind-solar hybrid power plants and distribution network.

CHAPTER 13

APPENDIX

SOURCE CODE:

app.py

```

from flask import Flask, render_template, url_for, redirect, flash, request
from flask_login import login_user, LoginManager, login_required, logout_user
from wtforms.validators import InputRequired, Length, ValidationError
from flask_bcrypt import Bcrypt
from wtforms import StringField, PasswordField, SubmitField
from flask_wtf import FlaskForm
from flask_cors import CORS
import joblib
import users_db
import requests

# Enter your API key here
api_key = "9dc73970faafde4beb008b5e93ca7ab1"

# base_url variable to store url
base_url = "http://api.openweathermap.org/data/2.5/weather?"

# api key from my cloud account
API_KEY = "Ke3Pq9j_CPfm7SmFG5cJerK2XZ4J-AeEMDYrAlxlnA1"

token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

```

```
app = Flask(__name__)
CORS(app)
```

```
app = Flask(__name__)
bcrypt = Bcrypt(app)
app.config['SECRET_KEY'] = 'vidyaibm'
```

```
conn = users_db.connect("DATABASE=bludb;HOSTNAME=7ubf76c4-r56c-4ay0-72v9-
ac2b4345f4a4.c3n31cmd0nprnk39u98g.databases.appdomain.cloud;PORT=32286;
SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=ibq38740;PWD=
uX6v3MioF6cEBa7c", "", "")
```

```
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'
```

```
@login_manager.user_loader
def load_user(user_id):
    stmt = users_db.prepare(conn, 'SELECT * FROM user WHERE id=?')
    users_db.bind_param(stmt, 1, user_id)
    users_db.execute(stmt)
    user = users_db.fetch_tuple(stmt)
    usr_obj = User(user[0], user[1], user[2])
    return usr_obj
```

```
class User:
    def __init__(self, id, email, username):
        self.id = id
        self.username = username
        self.email = email
```

```
def to_json(self):
    return {"username": self.username, "email": self.email}
```

```
def is_authenticated(self):
    return True
```

```
def is_active(self):
    return True
```

```
def is_anonymous(self):
    return False
```

```
def get_id(self):
    return str(self.id)
```

```
class RegisterForm(FlaskForm):
    email = StringField(validators=[InputRequired(), Length(min=4, max=50)],
render_kw={"placeholder": "Email"})

    username = StringField(validators=[InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})

    rollnumber = StringField(validators=[InputRequired(), Length(min=5, max=10)],
render_kw={"placeholder": "RollNumber"})

    password = PasswordField(validators=[InputRequired(), Length(min=8,
max=20)],render_kw={"placeholder": "Password"})

    submit = SubmitField('Register')
```

```
def validate_username(self, username):
    stmt = users_db.prepare(conn, 'SELECT * FROM user WHERE username=?')
    users_db.bind_param(stmt, 1, username.data)
    users_db.execute(stmt)
```



```

existing_user_username = users_db.fetch_tuple(stmt)

if existing_user_username:

    raise ValidationError('That username already exists. Try another one.')


class LoginForm(FlaskForm):

    username = StringField(validators=[InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})

    password = PasswordField(validators=[InputRequired(), Length(min=8, max=20)],
render_kw={"placeholder": "Password"})

    submit = SubmitField('Login')


class UpdateForm(FlaskForm):

    username = StringField(validators=[InputRequired(), Length(min=4, max=20)],
render_kw={"placeholder": "Username"})

    oldpassword = PasswordField(validators=[InputRequired(), Length(min=8,
max=20)], render_kw={"placeholder": "Previous Password"})

    password = PasswordField(validators=[InputRequired(), Length(min=8, max=20)],
render_kw={"placeholder": "Password"})

    submit = SubmitField('Update')


@app.route('/')

def home():

    return render_template('home.html')


@app.route('/login', methods=['GET', 'POST'])

def login():

    form = LoginForm()

    if form.validate_on_submit():

        stmt = users_db.prepare(conn, 'SELECT * FROM user WHERE username=?')

        users_db.bind_param(stmt, 1, form.username.data)

        users_db.execute(stmt)

```

```

user = users_db.fetch_tuple(stmt)
if user:
    if bcrypt.check_password_hash(user[4], form.password.data):
        usr_obj = User(user[0], user[1], user[2])
        login_user(usr_obj)
        return redirect(url_for('welcome'))
    else:
        print('Hello')
        flash(f'Invalid credentials, check and try logging in again.', 'danger')
        return redirect(url_for('login'))
return render_template('login.html', form=form)

@app.route('/welcome', methods=['GET', 'POST'])
@login_required
def welcome():
    return render_template('welcome.html')

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

@app.route('/logout', methods=['GET', 'POST'])
@login_required
def logout():
    logout_user()
    return redirect(url_for('login'))

@app.route('/register', methods=['GET', 'POST'])
def register():
    form = RegisterForm()
    if form.validate_on_submit():

```

```

        hashed_password = bcrypt.generate_password_hash(form.password.data)
        stmt = ibm_db.prepare(conn, 'INSERT INTO user (email, username, roll_number,
pass_word) VALUES (?, ?, ?, ?)')
        users_db.bind_param(stmt, 1, form.email.data)
        users_db.bind_param(stmt, 2, form.username.data)
        users_db.bind_param(stmt, 4, hashed_password)
        #hash causes size to exceed VARCHAR size in DB2, hence made
VARCHAR(8000)
        users_db.execute(stmt)
        return redirect(url_for('login'))
    return render_template('register.html', form=form)

```

```

@app.route('/update', methods=['GET', 'POST'])
def update():
    form = UpdateForm()
    if form.validate_on_submit():
        stmt = users_db.prepare(conn, 'SELECT * FROM user WHERE username=?')
        users_db.bind_param(stmt, 1, form.username.data)
        users_db.execute(stmt)
        user = users_db.fetch_tuple(stmt)
        if user:
            if bcrypt.check_password_hash(user[4], form.oldpassword.data):
                print(user)
                hashed_password1 =
bcrypt.generate_password_hash(form.password.data)
                stmt = users_db.prepare(conn, 'UPDATE user SET pass_word=? WHERE
username=?')
                users_db.bind_param(stmt, 1, hashed_password1)
                users_db.bind_param(stmt, 2, form.username.data)
                user = users_db.execute(stmt)
                flash(f'Password changed successfully.', 'success')

```

```

        return redirect(url_for('home'))
    else:
        flash(f'Invalid password, Enter valid password.', 'danger')
        return redirect(url_for('update'))
    else:
        flash(f'Invalid user, Enter valid User.', 'danger')
        return redirect(url_for('update'))
return render_template('update.html', form=form)

```

```

@app.route('/predict', methods=['POST'])
def predictSpecies():
    hwl = float(request.form['hwl'])
    cn = input(request.form['cn'])
    # complete_url variable to store complete url address
    complete_url = base_url + "q=" + cn + "&appid=" + api_key
    # get method of requests module
    # return response object
    response = requests.get(complete_url)

    # json method of response object
    # convert json format data into python format data
    x = response.json()

    # Now x contains list of nested dictionaries
    # Check the value of "cod" key is equal to "404", means city is found otherwise,
    # city is not found
    if x["cod"] != "404":

        # store the value of "main"

```

key in variable y

y = x["main"]

store the value corresponding

to the "temp" key of y

current_temperature = y["temp"]

store the value corresponding

to the "pressure" key of y

current_pressure = y["pressure"]

store the value corresponding

to the "humidity" key of y

current_humidity = y["humidity"]

#store the value of "wind"

key in variable a

a = x["wind"]

store the value corresponding

to the "wind speed" key of a

current_windspeed = a["speed"]

current_winddeg = a["deg"]

store the value of "weather"

key in variable z

z = x["weather"]

store the value corresponding

to the "description" key at

```

# the 0th index of z
weather_description = z[0]["description"]

else:

    print(" City Not Found ")

    X = [[current_temperature, current_pressure, current_humidity, current_windspeed,
current_winddeg,hwl]]

    ## NOTE: manually define and pass the array(s) of values to be scored in the next
line

    payload_scoring = {"input_data": [{"field": [[current_temperature, current_pressure,
current_humidity, current_windspeed, current_winddeg,hwl]], "values": X}]}

    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/9c062b2f-991b-489c-956b-
93a8f76f45b5/predictions?version=2022-11-19',
json=payload_scoring,headers={'Authorization': 'Bearer ' + mltoken})

    print(response_scoring)

    predictions = response_scoring.json()

    predict = predictions['predictions'][0]['values'][0][0]

    print("Final prediction : ",predict)

    return render_template('predict.html',predict=predict)

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

```

dashboard.html

```

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

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

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

  <title>DashBoard</title>

</head>

<body>

  <h1>IBM-Predicting the energy output of wind turbine based on weather
condition</h1> <br>

  <h7>Dashboard</h7>

  <form method="POST" action="/predict">

    <p>Height of the Wind mill</p>

    <input name="hwl"required>

    <p>Location i.e City name</p>

    <input name="cn"required>

    <br>

    <br>

```

```

        <button type="submit">Evaluate</button>
    </form>
</body>
<br><br><br><br><br>
</html>

```

home.html

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css')
}}" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
</head>

<body>
    <div class="col-md-8">
        {% with messages = get_flashed_messages(with_categories=true) %}
            {% if messages %}
                {% for category, message in messages %}

```



```

        <div class="alert alert-{{category}}">
            {{ message }}
        </div>
    {% endfor %}
{% endif %}
{% endwith %}
{% block content %} {% endblock %}
</div>

<h1>IBM-Predicting the energy output of wind turbine based on weather
condition</h1> <br>
<br><br><br>
<a href="{{ url_for('login') }}">
    <button class="btn" type="submit" >Login</button>
</a>
<a href="{{ url_for('register') }}">
    <button class="btn" type="submit">Register</button>
</a>
<a href="{{ url_for('update') }}">
    <button class="btn" type="submit">Update Password</button>
</a>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdsJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1
" crossorigin="anonymous"></script>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-

```

```
JjSmVgyd0p3pXB1rRibZUAYolly6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

</body>

<br><br><br><br><br><br><br>

</html>
```

login.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

  <title>Login</title>

  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

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

</head>

<body>

  <div class="col-md-8">

    {% with messages = get_flashed_messages(with_categories=true) %}

      {% if messages %}

        {% for category, message in messages %}

          <div class="alert alert-{{category}}">

            {{ message }}


```

```

        </div>
        {% endfor %}
    {% endif %}
    {% endwith %}
    {% block content %} {% endblock %}
</div>

<h1>IBM-Predicting the energy output of wind turbine based on weather
condition</h1> <br>

<h7>Login Page</h7> <br>

<form method="POST" action="">
    {{ form.hidden_tag() }}
    <fieldset class="form-group">
        <div>
            {{ form.username.label(class="form-control-label") }}

            {% if form.username.errors %}
                {{ form.username(class="form-control form-control-lg is-invalid") }}
                <div class="invalid-feedback">
                    {% for error in form.username.errors %}
                        <span>{{ error }}</span>
                    {% endfor %}
                </div>
            {% else %}
                {{ form.username(class="form-control form-control-lg") }}
            {% endif %}
        </div>
    
```

```

<div>
  {{ form.password.label(class="form-control-label") }}

  {% if form.password.errors %}
    {{ form.password(class="form-control form-control-lg is-invalid") }}
    <div class="invalid-feedback">
      {% for error in form.password.errors %}
        <span>{{ error }}</span>
      {% endfor %}
    </div>
  {% else %}
    {{ form.password(class="form-control form-control-lg") }}
  {% endif %}
</div>

```

```

</fieldset>
<div class="form-group">
  {{ form.submit(class="btn btn-outline-info") }}
</div>

```

```

<small class="text-muted ml-2"><br>
  <a href='{{url_for('register')}}'>Do not have an account? Sign Up?</a>
</small>

```

```

</form>

```

```

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

```

```

<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"

```

```
integrity="sha384-
UO2eT0CpHqdSJK6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1
" crossorigin="anonymous"></script>
```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIly6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
```

```
</body>
```

```
<br><br><br><br><br>
```

```
</html>
```

predict.html

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
```

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

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

```
<title>DashBoard</title>
```

```
</head>
```

```
<body>
```

```
<h1>PREDICTION OF ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER
CONDITION</h1> <br>
```

```
<h1>The predicted value is</h1>
```

```

<h1>{{predict}}</h1>
<a href="{{url_for('dashboard')}}">
    <button class="btn" type="submit" >Go back</button> <br><br><br>
</a>
<a href="{{url_for('logout')}}">
    <button class="btn" type="submit" >Logout</button>
</a>
</body>
<br><br><br><br><br>
</html>

```

register.html

```

<!DOCTYPE html>
<html lang="en">

<head>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css')
}}" />
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Register</title>
</head>

<body>

```

```
<h1>Predicting the energy output of wind turbine based on weather
condition</h1> <br>
```

```
<h7>Register Page</h7> <br>
```

```
<form method="POST" action="">
```

```
{ form.hidden_tag() }}
```

```
<fieldset class="form-group">
```

```
<div>
```

```
  {{ form.email.label(class="form-control-label") }}
```

```
  {% if form.email.errors %}
```

```
    {{ form.email(class="form-control form-control-lg is-invalid") }}
```

```
    <div class="invalid-feedback">
```

```
      {% for error in form.email.errors %}
```

```
        <span>{{ error }}</span>
```

```
      {% endfor %}
```

```
    </div>
```

```
  {% else %}
```

```
    {{ form.email(class="form-control form-control-lg") }}
```

```
  {% endif %}
```

```
</div>
```

```
<div>
```

```
  {{ form.username.label(class="form-control-label") }}
```

```

{% if form.username.errors %}
    {{ form.username(class="form-control form-control-lg is-invalid") }}
    <div class="invalid-feedback">
        {% for error in form.username.errors %}
            <span>{{ error }}</span>
        {% endfor %}
    </div>
{% else %}
    {{ form.username(class="form-control form-control-lg") }}
{% endif %}
</div>

```

```

<div>
    {{ form.password.label(class="form-control-label") }}

    {% if form.password.errors %}
        {{ form.password(class="form-control form-control-lg is-invalid") }}
        <div class="invalid-feedback">
            {% for error in form.password.errors %}
                <span>{{ error }}</span>
            {% endfor %}
        </div>
    {% else %}
        {{ form.password(class="form-control form-control-lg") }}
    {% endif %}
</div>

```

```

</fieldset>

```

```

<div class="form-group">

```



```

    {{ form.submit(class="btn btn-outline-info") }}
</div>

<small class="text-muted ml-2"><br><br>
    <a href="{{ url_for('login') }}">Already have an account? Log In</a>
</small>

</form>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJK6hJty5KVphtPhzWj9W01clHTMGa3JDZwrnQq4sF86dIHNDz0W1
" crossorigin="anonymous"></script>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIly6OrQ6VrjIEaFf/nJGzlxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

</body>

<br><br><br><br><br>

</html>

```

update.html

```

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

```

```

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

<title>Login</title>

<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

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

</head>

<body>

<div class="col-md-8">

    {% with messages = get_flashed_messages(with_categories=true) %}
        {% if messages %}
            {% for category, message in messages %}
                <div class="alert alert-{{category}}">
                    {{ message }}
                </div>
            {% endfor %}
        {% endif %}
    {% endwith %}

    {% block content %} {% endblock %}

</div>

<h1>IBM-Predicting the energy output of wind turbine based on weather
condition</h1> <br>

<h2>Update Password</h2> <br>

<form method="POST" action="">
    {{ form.hidden_tag() }}
    <fieldset class="form-group">
        <div>

```

```

{{ form.username.label(class="form-control-label") }}

{% if form.username.errors %}
    {{ form.username(class="form-control form-control-lg is-invalid") }}
    <div class="invalid-feedback">
        {% for error in form.username.errors %}
            <span>{{ error }}</span>
        {% endfor %}
    </div>
{% else %}
    {{ form.username(class="form-control form-control-lg") }}
{% endif %}
</div>

<div>
    {{ form.oldpassword.label(class="form-control-label") }}

    {% if form.oldpassword.errors %}
        {{ form.oldpassword(class="form-control form-control-lg is-invalid") }}
        <div class="invalid-feedback">
            {% for error in form.oldpassword.errors %}
                <span>{{ error }}</span>
            {% endfor %}
        </div>
    {% else %}
        {{ form.oldpassword(class="form-control form-control-lg") }}
    {% endif %}
</div>

```

```

<div>
  {{ form.password.label(class="form-control-label") }}

  {% if form.password.errors %}
    {{ form.password(class="form-control form-control-lg is-invalid") }}
    <div class="invalid-feedback">
      {% for error in form.password.errors %}
        <span>{{ error }}</span>
      {% endfor %}
    </div>
  {% else %}
    {{ form.password(class="form-control form-control-lg") }}
  {% endif %}
</div>

```

```

</fieldset>

```

```

<div class="form-group">
  {{ form.submit(class="btn btn-outline-info") }}
</div>

```

```

</form>

```

```

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

```

```

<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdsJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1
" crossorigin="anonymous"></script>

```

```

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-

```

```
JjSmVgyd0p3pXB1rRibZUAYolly6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

</body>

<br><br><br><br><br>

</html>
```

welcome.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

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

  <meta charset="UTF-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

  <title>Welcome Page</title>

</head>

<body>

  <h1>IBM-Predicting the energy output of wind turbine based on weather
condition</h1> <br>

  <h7>Welcome!</h7> <br>

  <a href="{{url_for('dashboard')}}">

    <button class="btn" type="submit" >Dashboard</button>

  </a>

  <a href="{{url_for('logout')}}">
```

```
<button class="btn" type="submit" >Logout</button>
```

```
</a>
```

Wind power or wind energy is mostly the use of wind turbines to generate electricity. Wind power is a popular, sustainable, renewable energy source that has a much smaller impact on the environment than burning fossil fuels. Historically, wind power has been used in sails, windmills and windpumps but today it is mostly used to generate electricity. Wind farms consist of many individual wind turbines, which are connected to the electric power transmission network. Wind energy conversion devices like wind turbines are used for converting wind energy into mechanical energy. Wind turbines consist basically of a few sails, vanes or blades radiating from a central axis. When wind blows against the blades or vanes, they rotate about the axis. This rotational motion is utilised to perform some useful work. By connecting the wind turbine to an electric generator wind energy can be converted into electric energy.

In India, winds are relatively low (5 km/hr to 15/20 km/hr) and vary appreciably with season. This makes the exploitation of wind energy an expensive project.

```
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
```

```
<script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdsJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1
" crossorigin="anonymous"></script>
```

```
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIlly6OrQ6VrjIEaFf/nJGzlxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
```

```
</body><br><br><br><br><br>
```

```
</html>
```

style.css

```
.header {
padding: 5px 120px;
```

```
width: 150px;
height: 70px;
background-color: #89238e;
}
footer{
background-color: #06182b;
color: #ffffff;
text-align: center;
font-size: 20px;
}
.border {
padding: 80px 50px;
width: 400px;
height: 450px;
border: 1px solid #8e238e;
border-radius: 0px;
background-color: #cd9ac8;
}

.btn {
padding: 10px 40px;
background-color: #8e2380;
color: #ffffff;
font-style: oblique;
font-weight: bold;
border-radius: 10px;
text-align: center;
}

.textbox {
```

```
padding: 10px 40px;  
background-color: #236b8e;  
text-shadow: #ffffff;  
border-radius: 10px;  
}
```

```
::placeholder {  
  color: #ffffff;  
  opacity: 1;  
  font-style: oblique;  
  font-weight: bold;  
}
```

```
.word {  
  color: #ffffff;  
  font-style: oblique;  
  font-weight: bold;  
}
```

```
.bottom {  
  color: #236b8e;  
  font-style: oblique;  
  font-weight: bold;  
}
```

```
body {  
  background: #fafafa;  
  color: #ffffff;  
  margin-top: 5rem;  
  background-image: url('https://unsplash.com/photos/NNn2Dc6niVU');
```



```
background-repeat: no-repeat;  
background-attachment: fixed;  
background-size: cover;  
text-align: center;  
}
```

```
h1, h2, h3, h4, h5, h6 {
```

```
padding: 30px;  
text-align: center;  
font-size: 40px;  
color: white;  
}
```

```
h7{  
padding: 30px;  
text-align: center;  
font-size: 30px;  
color: white;  
}
```

```
p{  
padding: 40px;  
color: white;  
text-align: center;  
}
```

```
.bg-steel {  
background-color: #8a5f7f;  
}
```

```
.site-header .navbar-nav .nav-link {  
color: #dbcbd5;
```

```
}
```

```
.site-header .navbar-nav .nav-link:hover {  
  color: #ffffff;  
}
```

```
.site-header .navbar-nav .nav-link.active {  
  font-weight: 500;  
}
```

```
.content-section {  
  background: #ffffff;  
  padding: 10px 20px;  
  border: 1px solid #dddddd;  
  border-radius: 3px;  
  margin-bottom: 20px;  
}
```

```
div{  
  text-align: center;  
  display: inline-block;  
  
}
```

```
.btn {  
  outline: none;  
  border: none;  
  cursor: pointer;  
  display: inline-block;  
  margin: 0 auto;  
  padding: 0.9rem 2.5rem;
```

```
text-align: center;
background-color: white;
color: #2B547E;
border-radius: 4px;
box-shadow: 0px 3px 6px rgba(0, 0, 0, 0.16);
}
```

```
.article-title {
  color: #444444;
}
```

```
a.article-title:hover {
  color: #b842ca;
  text-decoration: none;
}
```

```
.article-content {
  white-space: pre-line;
}
```

```
.article-img {
  height: 65px;
  width: 65px;
  margin-right: 16px;
}
```

```
.article-metadata {
  padding-bottom: 1px;
  margin-bottom: 4px;
  border-bottom: 1px solid #e3e3e3
}
```

```
}
```

```
.article-metadata a:hover {
  color: #333;
  text-decoration: none;
}
```

```
.article-svg {
  width: 25px;
  height: 25px;
  vertical-align: middle;
}
```

```
.account-img {
  height: 125px;
  width: 125px;
  margin-right: 20px;
  margin-bottom: 16px;
}
```

```
.account-heading {
  font-size: 2.5rem;
}
```

notificationsystem.py

```
# importing required modules and libraries
import datetime # to read present date
import time # to suspend the execution for a specific time
import requests # to retrieve COVID stats from web
```

```

from plyer import notification # to get notification on the computer

# initializing a variable with None (temporary)
# indicating that there is no data available currently
from app.py import predict
user = int(input("Enter 1 to send notification, enter 0 to not send a notification"))
demandpower = float(input("Enter demanded value"))
title = "Sorry, demanded power can't be generated for 2022 -11-19"
message = "Predicted value is {predict} but, Demanded value is {demandpower}"
    # repeating the loop for multiple times
while(user ==1):
    notification.notify(
        # defining the title of the notification,
        title = title,

        # defining the message of the notification
        message = message,
        # creating icon for the notification
        # we have to download a icon of ico file format
        app_icon = "windmill.ico",
        # the notification stays for 30 seconds
        timeout = 30,

    )
break

```

SCREENSHOTS:

DATASET

Wind speed at height of 30 meters (m/s)	Wind direction at height of 30 meters (°)	Wind speed at height of 50 meters (m/s)	Wind direction at height of 50 meters (°)	Wind speed - at the height of wheel hub (m/s)	Wind speed - at the height of wheel hub (°)	Air temperature (°C)	Atmosphere (hpa)	Relative humidity (%)	Power (MW)
1.991	74.814	2.094	77.667	2.494	74.5	-13.484	889.9	76.32	0.254383
1.698	75.048	1.757	88.733	1.882	74.367	-13.691	889.6	76.757	0.329703
2.313	84.688	2.344	89.1	2.35	89	-13.766	889.9	76.981	0.296306
2.494	74.939	2.574	87.267	2.808	82.733	-13.691	889.7	76.821	0.187590
2.192	91.14	2.558	96.9	2.924	92.967	-13.447	890.0	74.571	0.081005
1.98	91.921	2.049	97	2.137	100.133	-13.983	889.7	71.808	0.189721
2.336	98.886	2.408	110.533	2.506	103	-14.014	890.0	70.816	0.265041
2.287	98.843	2.507	104.5	2.868	103.7	-14.149	889.5	69.525	0.241593
2.407	84.094	2.541	94.3	2.657	100	-14.116	890.0	69.824	0.184037
2.718	65.815	2.781	76.333	2.841	82.167	-14.183	889.8	75.029	0.119375
2.957	79.623	2.737	91.3	1.968	85.633	-14.166	889.5	77.312	0.301280
2.113	98.282	1.753	105.3	2.203	103.467	-14.149	890.1	77.035	0.508055
2.352	99.336	2.099	104.2	2.332	103.9	-14.266	889.7	74.261	1.082193
2.137	100.764	1.811	108.8	1.957	102.967	-14.099	889.4	71.36	0.537188
2.036	98.808	1.884	106.2	1.868	99.7	-14.116	888.7	69.643	0.156325
2.012	100.042	1.784	105.567	2.227	100.5	-13.899	889.95	68.501	0.724778
1.317	98.583	1.229	101.6	1.555	107.2	-14.233	890.483	67.467	0.226671
1.977	77.711	1.923	79	1.822	80	-14.249	889.858	67.328	0.218855
1.633	66.143	1.793	72	1.692	74.667	-14.133	890.133	68.075	0.366652
1.792	91.502	1.917	99	2.013	84.433	-14.278	889.933	68.384	0.333966
1.744	92.199	1.678	101.133	2.049	88.4	-14.641	890.217	68.011	0.328282
2.274	75.561	2.444	82	2.163	88.5	-14.828	889.917	69.024	0.301991
3.04	91.358	2.721	101.367	3.054	104.033	-14.599	890.258	69.717	0.297017
1.588	67.462	1.987	74.733	2.144	73.667	-14.481	890.033	70.784	0.267173
1.699	71.661	2.187	79	2.567	72.333	-14.433	889.85	71.168	0.153482
2.056	34.496	2.297	46.7	2.348	57.4	-14.724	889.842	71.189	0.266463
1.677	46.142	2.186	56.833	2.579	56.167	-14.474	889.8	70.4	0.049740
2.332	94.707	2.974	99.7	3.361	90.567	-14.291	889.975	68.48	0.674328
1.073	105.682	1.806	97.7	2.432	94.733	-14.266	889.767	66.549	1.227149
1.313	96.23	1.662	96.533	1.908	95.867	-13.841	889.75	64.821	0.508766
2.717	102.112	2.784	106.233	3.076	102.467	-13.983	889.9	63.947	0.953580

IBM API KEY

API keys

Create, view, and work with API keys that you have access to manage. IBM Cloud API keys are associated with a user's identity and can be used to access cloud platform and classic infrastructure APIs, depending on the access that is assigned to the user. The following table displays a list of API keys created in this account. [Learn more.](#)

Looking for more options to manage API Keys? Try [IBM Cloud® Secrets Manager](#) for creating and leasing API keys dynamically and storing them securely in your own dedicated instance.

View: My IBM Cloud API keys

API keys associated with a user's identity have the same access that the user is assigned across all accounts. To update the access for an API key, assign or remove access for the user.

Status	Name	Description	Date Created
	PREDICTION	WIND POWER PREDICTION	2022-11-19 06:27 GMT

Items per page: 25 | 1-25 items | Page 1

IBM API KEY DETAILS

API key details


Name

PREDICTION

Description

WIND POWER PREDICTION

ID

ApiKey-01cbe9fc-ed7f-46fc-a225-b4c2559b7136

Status

Unlocked

Email

19csvidhyambikasr@drngpit.ac.in

Created by

Vidhyambika SR

Date created

2022-11-19 06:27 GMT

OPEN WEATHER API KEY

OpenWeather Weather in your city Guide API Dashboard Marketplace Pricing Maps Our Initiatives Partners Blog For Business Vidh... Support

New Products Services **API keys** Billing plans Payments Block logs My orders My profile Ask a question

You can generate as many API keys as needed for your subscription. We accumulate the total load from all of them.

Key	Name	Status	Actions
f673430834ae0300e227899836f9079a	Default	Active	🔍 ✎ ✕
9dc73970faafde4beb00b5e93ca7ab1	Current Weather	Active	🔍 ✎ ✕

Create key

API key name **Generate**

Product Collections
 Current and Forecast APIs
 Historical Weather Data
 Weather Maps
 Weather Dashboard

Subscription
 How to start
 Pricing
 Subscribe for free
 FAQ

Company
 OpenWeather is a team of IT experts and data scientists that has been practising deep weather data science since 2014. For each point on the globe, OpenWeather provides historical, current and forecasted weather data via light-speed APIs. Headquarters in London, UK.

IBM MODEL DEPLOYMENT

The screenshot displays the IBM Watson Studio interface for a deployed model named 'wind_power_prediction'. The model is in a 'Deployed' state and is 'Online'. The interface includes an 'API reference' tab and a 'Test' tab. The 'API reference' tab shows the 'Direct link' endpoint: `https://us-south.ml.cloud.ibm.com/ml/v4/deployments/9c062b2f-991b-489c-956b-93a8f76f45b5/predictions?version=...`. Below this, there are 'Code snippets' for various languages: cURL, Java, JavaScript, Python, and Scala. The cURL snippet includes a note about setting the API key and a detailed curl command for making a prediction request. On the right side, a sidebar provides metadata for the deployment, including the creation and update timestamps (Nov 19, 2022, 1:07 PM), the deployment ID (9c062b2f-991b-489c-956b-93...), the software specification (runtime-22.1-py3.9), the number of copies (1), the serving name (No serving name), the description (No description provided), tags (Add tags to make assets easier to find), and the associated asset (windpower).

PREDICTION IN WEBAPP

The screenshot shows a web application dashboard titled 'IBM-PREDICTING ENERGY OUTPUT OF WIND TURBINE BASED ON THE WEATHER CONDITION'. The dashboard has a header with the title and a 'Dashboard' label. Below the header, there is a 'Welcome, Admin' message. The main content area features a background image of three wind turbines in a field. On the left side, there are two input fields: 'Height of the Wind mill' and 'Location i.e City name'. Below these fields is a 'Submit' button.

PREDICTION ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER CONDITION



The predicted value is 63.7 MW

Using Random Forest Regressor

[Go back](#) [Logout](#)

OUTPUT OF WEATHERCHECK.PY (FEATURE 1)

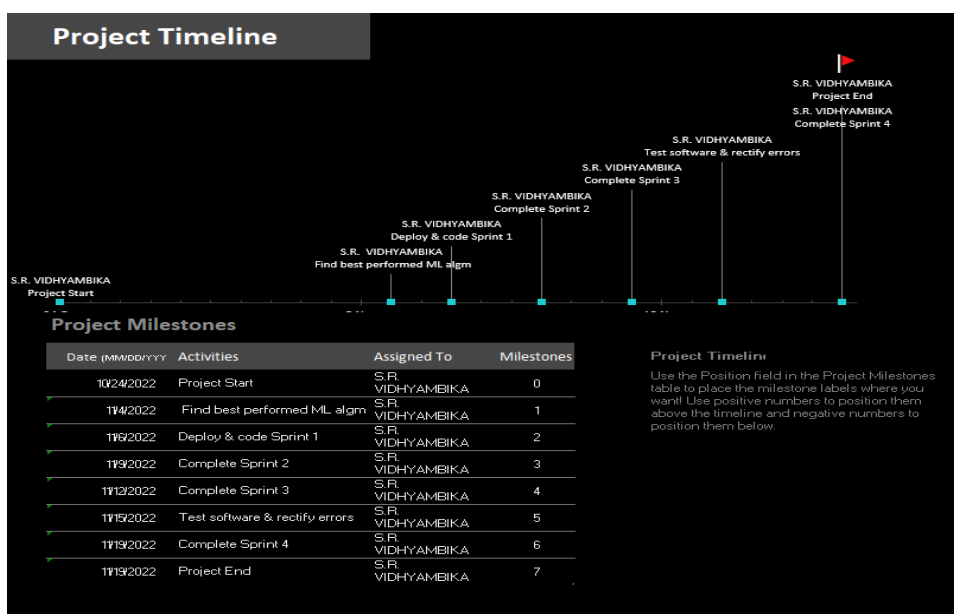
```

PROBLEMS 52 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Try the new cross-platform PowerShell https://aka.ms/powershell

PS D:\sprint1> & 'C:\Users\VIDHYAMBIKA S.R\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\VIDHYAMBIKA S.R\vscode\extensions\ms-p
ython.python-2022.18.2\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '54153' '--' 'd:\sprint1\weathercheck.py'
Enter city name : chennai
Temperature (in kelvin unit) = 297.14
atmospheric pressure (in hPa unit) = 1008
humidity (in percentage) = 83
description = mist
wind speed (in meter per sec) = 4.63
wind direction (in degrees) = 20
PS D:\sprint1>

```

MILESTONE AND ACTIVITY LIST



IMPORTANT LINKS

GitHub link : [Final Deliverables](#)

Demo Video link:

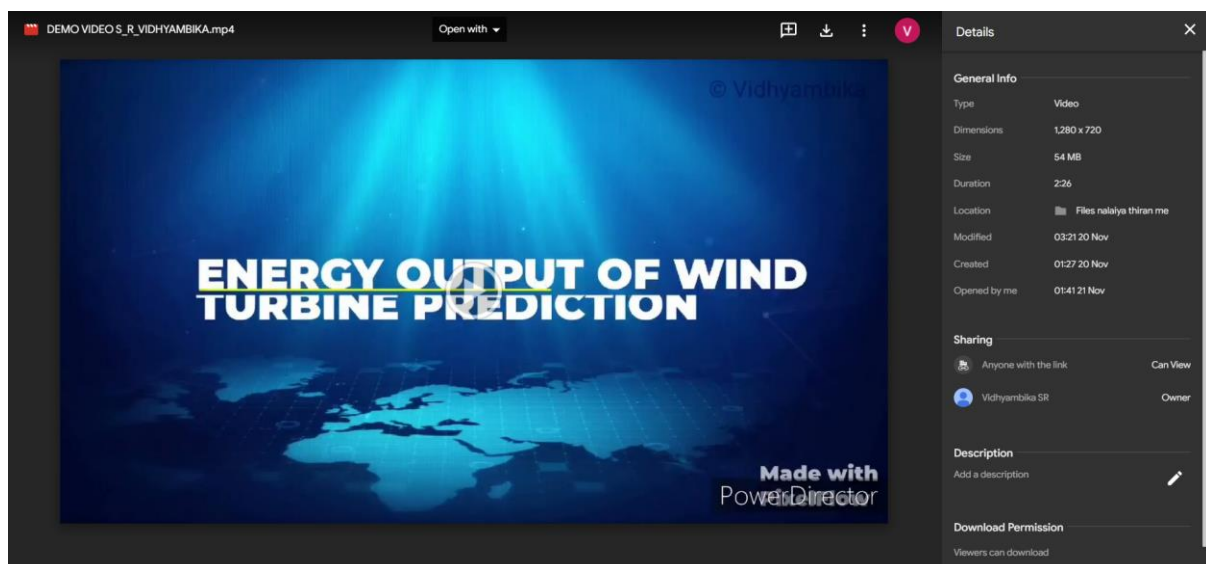
<https://drive.google.com/file/d/17PAbWtoQTSSJAgJu7L4IlwcVW6YNEvHT/view?usp=sharing>

MY IBM API KEY : Ke3Pq9j_CPfm7SmFG5cJerK2XZ4J-AeEMDYrAlxIxnA1

MY OPEN WEATHER MAP API KEY: 9dc73970faafde4beb008b5e93ca7ab1

DEMO VIDEO UPLOADED DATE : 20-11-2022

DEMO VIDEO UPLOADED TIME : 1.27 AM



MADE BY S.R. VIDHYAMBIKA