**Increment model**

Hu22csen0100300

P.Sunanditha

Incremental Model Stages:

1. Requirements Gathering

2. Design

3. Implementation (Increment 1)

4. Testing and Integration (Increment 1)

5. Implementation (Increment 2)

6. Testing and Integration (Increment 2)

7. Deployment

8. Maintenance

Stage 1: Requirements Gathering

- Gather requirements from stakeholders

- Define the input variables: temperature, humidity, wind speed

- Define the output variable: weather prediction

Stage 2: Design

- Design the weather modeling system using a quadratic equation

- Define the coefficients: a = 0.5, b = -0.2, c = 0.1, d = -15

Stage 3: Implementation (Increment 1)

- Implement the quadratic equation in Python

- Define the input variables: temperature, humidity, wind speed

- Define the output variable: weather prediction

# Increment 1

def weather\_prediction(temperature, humidity, wind\_speed):

prediction = 0.5 \* temperature\*\*2 - 0.2 \* humidity + 0.1 \* wind\_speed - 15

return prediction

# Increment 1: Weather Prediction Function

def weather\_prediction(temperature, humidity, wind\_speed):

prediction = 0.5 \* temperature\*\*2 - 0.2 \* humidity + 0.1 \* wind\_speed - 15

return prediction

# Test the function

temperature = 20

humidity = 60

wind\_speed = 10

prediction = weather\_prediction(temperature, humidity, wind\_speed)

print("Weather Prediction:", prediction)

Stage 4: Testing and Integration (Increment 1)

- Test the weather prediction function with sample inputs

- Integrate the function with a user interface (e.g., command-line interface)

# Testing and Integration

temperature = 20

humidity = 60

wind\_speed = 10

prediction = weather\_prediction(temperature, humidity, wind\_speed)

print("Weather Prediction:", prediction)

Stage 5: Implementation (Increment 2)

- Implement a graphical user interface (GUI) using a library like Tkinter or PyQt

- Integrate the GUI with the weather prediction function

# Increment 2

import ipywidgets as widgets

from IPython.display import display

# Function to calculate weather prediction

def weather\_prediction(T, H, W):

return 0.5 \* T\*\*2 - 0.2 \* H + 0.1 \* W - 15

# Create input widgets

temperature\_input = widgets.FloatText(description="Temperature (°C):", value=0)

humidity\_input = widgets.FloatText(description="Humidity (%):", value=0)

wind\_speed\_input = widgets.FloatText(description="Wind Speed (m/s):", value=0)

predict\_button = widgets.Button(description="Predict")

output\_label = widgets.Label(value="Weather Prediction: ")

values\_label = widgets.Label(value="")

# Function to predict weather

def predict\_weather(button):

try:

# Get input values

temperature = temperature\_input.value

humidity = humidity\_input.value

wind\_speed = wind\_speed\_input.value

# Check for valid ranges

if temperature < -50 or temperature > 50:

output\_label.value = "Temperature out of range (-50 to 50)."

return

if humidity < 0 or humidity > 100:

output\_label.value = "Humidity out of range (0 to 100)."

return

if wind\_speed < 0:

output\_label.value = "Wind speed cannot be negative."

return

# Calculate prediction

prediction = weather\_prediction(temperature, humidity, wind\_speed)

# Update output labels

output\_label.value = f"Weather Prediction: {prediction:.2f}"

values\_label.value = f"Input Values - Temp: {temperature}°C, Humidity: {humidity}%, Wind: {wind\_speed} m/s"

except ValueError:

output\_label.value = "Invalid input! Please enter numbers."

# Attach event

predict\_button.on\_click(predict\_weather)

# Display widgets

display(temperature\_input, humidity\_input, wind\_speed\_input, predict\_button, output\_label, values\_label)

Stage 6: Testing and Integration (Increment 2)

- Test the GUI application with sample inputs

- Integrate the GUI with the weather prediction function

Stage 7: Deployment

- Deploy the GUI application to a production environment

- Make the application available to users

Stage 8: Maintenance

- Monitor the application's performance and accuracy

- Update the coefficients and weather prediction logic as needed

- Fix any bugs or issues that arise