

INTERNSHIP REPORT

ON

Personalized Weather-Health Assistant: AI-Powered Weather-Based Health
Advisory System

A report submitted in partial fulfillment of the requirements for the award of the
degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence & Machine Learning)

Submitted by

M HIMA BINDU	21751A3354
M REDDY MOHITH	21751A3356
VANDAVASI KUMAR	21751A3394
V SUNEETHA	21751A3395

Under the Guidance of

Mr. Gaurav Kumar, M. Tech

Scientist/ Engineer, NARL



Duration: 15-01-2025 to 5-05-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(AI&ML)

**SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES
CHITTOOR, ANDHRA PRADESH, INDIA- 517502**

(AUTONOMOUS)

**(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuram)
Chittoor,A.P-517127**

2024-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(AI&ML)



CERTIFICATE

This is to certify that the dissertation entitled “Personalized Weather-Health Assistant: AI-Powered Weather-Based Health Advisory System” is a Bonafide work, carried out by M HIMA BINDU(21751A3354), M REDDY MOHITH(21751A3356), VANDAVASI KUMAR(21751A3394), V SUNEETHA(21751A3395), in partial fulfilment of requirements for the award of the degree of BACHELOR OF TECHNOLOGY with the specialization in COMPUTER SCIENCE AND ENGINEERING(AI&ML) , during the academic year 2021-2025.

GUIDE

Mr. Gaurav Kumar,Mtech
Scientist/Engineer,
NARL
Gadanki,
Pin code:517501

HEAD OF THE DEPARTMENT

Dr. S. VijayaKumar, M.Tech., Ph.D.,
Department of CSE(AI&ML),
Sreenivasa Institute of Technology and
Management Studies Chittoor, A.P.
Pin code:517127

ACKNOWLEDGEMENT

The first person We would like to express Our profound gratitude to us most esteemed and beloved guide, **Mr. GAURAV KUMAR, M.Tech**, Scientist/Engineer, NARL, Gadanki. He has been an unwavering source of guidance and support, addressing our doubts with great patience and clarity, while consistently monitoring we progress both physically and virtually. His mentorship, supervision, and encouragement have been invaluable throughout our project. His dedication and commitment to excellence have been truly inspiring. We're truly grateful to have the privilege of learning under his guidance.

We grateful to **DR. S. VIJAYAKUMAR, M. TECH., PH.D.**, Head of the Department of Computer Science and Engineering(AI&ML), Sreenivasa Institute of Technology and Management Studies Chittoor, for his continuous support.

We acknowledge our sincere thanks to **DR. N.VENKATACHALAPATHI, M.T.ECH., PH.D.**, Principal of the College , Sreenivasa Institute of Technology and Management Studies Chittoor, for his kind cooperation and encouragement throughout this project.

Finally, we extend our deepest thanks to beloved parents and friends, whose love, support, and motivation have been our strength during the highs and lows of this journey. Their belief in us kept going and helped us successfully complete this project.

M HIMA BINDU	21751A3354
M REDDY MOHITH	21751A3356
VANDAVASI KUMAR	21751A3394
V SUNEETHA	21751A3395

DECLARATION

We are **M Hima Bindu(21751A3354)**, **M Reddy Mohith(21751A3356)**, **Vandavasi Kumar(21751A3394)**, **V Suneetha(21751A3395)** declare that this B.TECH project entitled " **Personalized Weather-Health Assistant: AI-Powered Weather-Based Health Advisor System** ", submitted to the Department of Computer Science and Engineering (AI&ML), Sreenivasa Institute of Technology and Management Studies Chittoor, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology, we carried out independently by us under the guidance of **Mr. GAURAV KUMAR, M.Tech, Scientist/Engineer, NARL, Gadanki.**

We further declare that this work has not been submitted earlier, either in part or full, for the award of any degree or diploma.

M HIMA BINDU	21751A3354
---------------------	-------------------

M REDDY MOHITH	21751A3356
-----------------------	-------------------

VANDAVASI KUMAR	21751A3394
------------------------	-------------------

V SUNEETHA	21751A3395
-------------------	-------------------

ABSTRACT

This project is about building a smart app that helps people stay safe from health problems caused by weather. Many people around the world have health issues like asthma, migraines, or heat sensitivity. Sometimes, weather changes like too much heat, bad air, or high humidity can make these problems worse. Our app, called SiHeal, gives early alerts to users when the weather may affect their health. The app works in a smart way. It first asks the user for their name, birth date, location, and the health conditions they have. Then, the app checks live weather data using a temperature API. It looks at things like temperature, air quality (AQI), and humidity. These details are matched with the user's health conditions. If the weather is getting dangerous, the app will send a warning to the user. This helps them take care of their health before things get worse. We trained a machine learning model using real data that links weather and health. The backend uses Python with a Random Forest model to predict the health risk. The frontend is made using React. We also used Flask and ngrok to connect everything online. The system stores all user data safely and uses it to make the model better in the future. This project is very helpful for people who want to take care of their health during changing weather. It works like a weather doctor that gives alerts just in time. Our goal is to save lives by sending helpful health tips and weather alerts on time. This app can be improved by adding more health conditions, using smart watch data, and giving voice alerts. SiHeal is a small but smart step toward a healthy future with the help of weather data and technology.

Table of contents

Chapter No.	Title	Page No
01	Introduction	1
	1.1 Company Profile	1
	1.2 Industry Profile	1
	1.3 Main Goal	2
02	Literature Survey	4
03	Project Description	9
	3.1 Problem Statement	9
	3.2 Dataset	11
	3.3 Architecture	14
	3.4 Evaluation Metrics	17
	3.5 Machine Learning Model	18
	3.6 Objective	22
4	Result and Discussion	23
	4.1 Experimentation Details	23
	4.2 Model Evaluation	24
	4.3 Discussion	28
05	Conclusion	30
	5.1 Conclusion	30
	4.2 Future Work	30
	References	32
	Appendix 1	34

CHAPTER 1

INTRODUCTION

1.1 Company Profile

The National Atmospheric Research Laboratory (NARL) is a leading research center located in Gadanki, a small village in Andhra Pradesh, India. It was set up by the Department of Space, Government of India, to study different layers of the atmosphere and understand how the air and weather behave. NARL mainly focuses on important work like studying rainfall, thunderstorms, wind movements, and how pollution spreads in the air.

Scientists at NARL use advanced tools like Doppler Weather Radars, Lidar systems, and weather balloons to collect information about the sky. They also work with international organizations to share knowledge and make better discoveries. The research done at NARL helps many fields like farming, communication, and public health. It plays a big role in helping us understand how changes in the weather affect our daily lives.

NARL's mission is not just to do research but also to support India's space and weather programs by giving accurate atmospheric information. In short, NARL is helping the world by making it safer and better prepared for future weather challenges.

1.2 Industry Profile

The weather and health monitoring industry is growing very fast today. It brings together two important areas: weather forecasting and health risk prediction. Many companies and research labs are trying to understand how changes in weather, like temperature, humidity, and pollution, can affect human health. This industry focuses on predicting the weather, checking air quality, and giving health alerts to people who are at risk. It uses smart devices like apps and wearable gadgets (such as smartwatches) to send real-time warnings when weather conditions become dangerous for people with health conditions like asthma, bronchitis, or heart disease.

The need for this industry has grown because of climate change, which is causing more extreme weather events, and air pollution, which is getting worse in many parts of the world. Reports by the World Health Organization (WHO) have shown that bad air and extreme weather

are now among the top reasons for poor health around the globe [1]. In recent years, the relationship between weather patterns and human health has become an increasingly important area of study. Sudden changes in temperature, humidity, and air quality can have serious effects on people, especially those with pre-existing medical conditions such as asthma, heart disease, or respiratory issues. For example, extreme heat can lead to heat strokes, dehydration, or even death if not addressed early. Similarly, poor air quality due to pollution can trigger asthma attacks or worsen lung conditions. With the growing impacts of climate change, these environmental factors are becoming more unpredictable and dangerous. As a result, there is a growing need for technology that can help people monitor and respond to these changing conditions in real time.

1.3 Main Goal

The main goal of this project is to develop a backend system that can analyze current environmental data, like temperature, humidity, and air quality, and predict health risk to individuals with certain health condition based on weather conditions. Based on this prediction, the system will generate health risk alerts that can be sent to users through mobile apps or wearable devices. These alerts can help people take preventive actions, such as staying indoors during high pollution levels or drinking more water during heat waves, potentially saving lives and improving overall public health.

To build this system, we use a synthetic dataset containing various weather conditions along with associated health risks. By applying random forest algorithm, the system learns patterns in the data and can predict risks based on real-time environmental inputs. For example, if the system detects a combination of high temperature and low humidity, conditions known to increase the risk of heatstroke. It will generate an alert warning users about this risk.

The concept behind this project is inspired by recent advancements in several fields. Researchers have shown that artificial intelligence, especially machine learning, can be used to improve early warning systems for weather-related health conditions [1]. Datasets like EpiClim provide detailed information about disease trends and climate data across different regions, which help in training accurate prediction models [2]. Wearable devices, such as smartwatches and fitness bands, can continuously collect health data and are increasingly being used to track

health status in real time [4][5]. The code of this project can be found at [**https://github.com/vandavasi-kumar/siheal**](https://github.com/vandavasi-kumar/siheal).

CHAPTER 2

LITERATURE SURVEY

Year	Title	Key Contributions	Project Relevance	Limitation	Domain
2021	Deep Learning in Human Activity Recognition with Wearable Sensors: A Review on Advances[4]	Shows how deep learning (smart computer programs) helps devices like smartwatches recognize human activities (walking, sitting, sleeping) more accurately.	Helps understand how smart models can use sensor data for predicting activities, which is important for tracking health risks.	Deep learning models need a lot of battery power and lots of different data to work perfectly.	Human Activity Recognition, Deep Learning, Wearable Technology
2021	Wearable Devices in Health Monitoring from the Environmental Towards Multiple Domains: A Survey[9]	Talks about how wearable devices watch not just body health but also the environment (air, temperature, etc.).	Useful because your project also looks at temperature and environmental conditions to predict health risks.	Environmental data is tricky; many factors outside body sensors can cause confusion in results.	Health Monitoring, Environmental and Multi-Domain Applications of Wearables
2022	What Role Could Wearable Tech Play in the Future of Mental Health Care?[10]	Explains how wearables can track mental health by noticing changes in sleep, heart rate, and behavior.	Shows how health tracking can be expanded beyond physical illness to emotional health too, giving broader ideas for alerts.	Mental health is complex, and devices might miss hidden emotional issues without talking to a person.	Mental Health, Wearable Technology

2023	A Survey on Smart Wearable Devices for Healthcare Applications[5]	Reviews how wearables help in remote health monitoring and explains how they work with wireless tech.	Very relevant because your app uses wearable data and remote updates to help users track health based on the environment.	Devices need strong security and privacy; user trust is easily lost if data isn't protected.	Wearable Technology, Healthcare Applications
2023	IoT and Health Monitoring Wearable Devices as Enabling Technologies for Sustainable Enhancement of Life Quality in Smart Environments [8]	Talks about how wearable devices and smart environments (like smart homes) can work together to improve health.	Supports your idea of connecting user data with real-time environment data to predict health risks.	IoT setups can be costly and complicated to set up and maintain.	IoT in Healthcare, Smart Environments, Wearable Devices
2023	Federated Learning and Blockchain-Enabled Fog-IoT Platform for Wearables in Predictive Healthcare[3]	Introduces privacy-protecting methods (Federated Learning + Blockchain) to collect wearable health data securely.	Helps in thinking about how to keep user health data private and safe in your project.	Setting up blockchain and federated learning can be heavy for small mobile devices.	Predictive Healthcare, Federated Learning, Blockchain, IoT
2024	Emerging Intelligent Wearable Devices for Cardiovascular Health Monitoring[6]	Shows how smart devices keep checking heart health all the time and alert early if problems happen.	Helps to understand how continuous monitoring can catch health risks early, matching your	Most focus is only on heart health; other health problems need different sensors and systems.	Biomedical Engineering, Wearable Devices, Cardiovascular Health

			goal of predictive alerts.		
2025	Reshaping the Healthcare World by AI-Integrated Wearable Sensors Following COVID-19[7]	Shows how AI and wearables got even better after COVID-19, helping track disease signs early without hospital visits.	Directly supports your idea of using AI models to predict health risks at home based on environment and body signs.	AI models need careful training; bad data can lead to wrong alerts or missed risks.	Healthcare Technology, Artificial Intelligence, Wearable Sensors, Post-COVID Health Monitoring
2025	EpiClim: Weekly District-Wise All-India Multi-Epidemics Climate-Health Dataset[2]	Gives a huge dataset that connects weather and disease outbreaks across India, weekly and by district.	It could be used to improve or validate your model by using similar climate-health patterns.	The dataset may not always match perfectly with your user's local conditions if not updated often.	Epidemiology, Climate and Health Data Science
2025	Optimizing Heat Alert Issuance with Reinforcement Learning[1]	Uses smart learning methods to decide when to send heat alerts that help lower hospital visits during heatwaves.	Matches perfectly because your project also needs to send smart alerts when temperature risks rise.	Reinforcement learning takes time to learn and needs a lot of historical data to be accurate.	Environmental Monitoring, Artificial Intelligence (Reinforcement Learning), Public Health

This literature survey reviews recent advancements in wearable health monitoring technologies, with a focus on how artificial intelligence (AI), environmental sensing, and remote data collection are reshaping healthcare. Starting from 2021[1], the field has evolved from using deep learning for recognizing human activity through sensor data to integrating complex AI systems capable of predicting health risks based on environmental and behavioral patterns. Early works emphasized the accuracy of deep learning models in identifying physical activities like walking or sleeping, though they also highlighted challenges such as high power consumption

and the need for diverse datasets. As research progressed into 2022[5] and 2023[8], wearables began to expand their scope beyond physical metrics, incorporating mental health monitoring by tracking changes in sleep, heart rate, and mood. During this period, there was also significant emphasis on remote health applications, data privacy, and the integration of wearables with wireless networks and IoT-enabled smart environments.

Recent studies from 2024[6] and 2025[7] illustrate a growing reliance on AI-integrated wearables that can support continuous health monitoring, particularly for cardiovascular health, and use predictive algorithms to detect potential health issues without the need for hospital visits. These developments are especially relevant in the post-COVID-19 context, where remote, AI-driven healthcare has gained momentum[7]. Furthermore, datasets like EpiClim have emerged, providing climate-health data that can enhance predictive modeling by connecting environmental factors with disease patterns[2]. At the same time, advanced AI techniques like reinforcement learning are being explored to optimize alert systems, such as heat warnings, although these methods require large amounts of historical data and careful calibration[1]. Across all studies, recurring challenges include maintaining user data privacy, managing the technical complexity of systems like federated learning and blockchain, and ensuring that predictions remain accurate in diverse real-world conditions.

Federated learning frameworks like FedHealth show how wearable healthcare data can be used to train predictive models across devices without exposing private information, which supports our approach to privacy-preserving health alerts [11]. Lightweight federated systems such as FedHome demonstrate that personalized models can run efficiently on local devices like smartphones and wearables, which is key for delivering real-time alerts without heavy cloud dependence [12]. Studies that integrate federated learning with blockchain highlight secure data handling in IoT-based health systems, helping us ensure both privacy and trust in our smartwatch-app connection [13]. Surveys on federated learning in medical IoT provide a roadmap for building scalable, secure, and personalized healthcare platforms, which directly supports our system design [14]. Additionally, reviews on smart healthcare show that AI-enabled federated systems can deliver fast and condition-aware decisions, which is critical for our temperature-triggered notification mechanism [15]. The combination of fog computing and federated learning in predictive wearable platforms offers low-latency data processing at the

edge, helping us send timely health advice when users are exposed to risky weather conditions [16]. Regionally detailed datasets like EpiClim support the model training process by linking real weather variables with health outcomes, improving the accuracy of predictions in the Indian context [17]. Research that evaluates the health impact of climate change in India helps define clear environmental thresholds (like temperature or AQI) for different age groups and health conditions, aligning with our alert logic [18]. Emerging AI applications in digital wellness coaching show how personalized health advice can be generated from real-time context, which matches our goal of sending educational tips along with alerts [19]. Finally, blockchain-based federated systems for healthcare IoT ensure end-to-end security and privacy, strengthening the backend of our predictive model that runs between smartwatch data, mobile input, and live temperature APIs [20].

CHAPTER 3

PROJECT DESCRIPTION

3.1 Problem Statement

The increasing impact of climate change on public health has brought a growing need for intelligent systems that can detect and alert users about health risks associated with environmental conditions. Modern studies suggest that integrating climate data, wearable sensors, machine learning, and mobile communication technologies can significantly enhance early warnings for disease outbreaks and other health conditions. One of the major concerns today is how changing weather parameters—such as temperature, humidity, and air quality—can trigger or aggravate specific health issues like heat strokes, respiratory problems, and infectious diseases. Research in reinforcement learning has demonstrated the potential to optimize alerts for such conditions, particularly in heat-related events, by learning from historical data patterns and decision feedback over time [1].

Datasets like EpiClim, which include weekly, district-level data on multiple epidemics across India, help correlate climate trends with disease occurrences [2]. These datasets, when analyzed with predictive algorithms, can act as the backbone for disease forecasting systems. Meanwhile, advancements in federated learning and blockchain have allowed health data from wearable devices to be securely processed in decentralized systems, ensuring privacy while still supporting large-scale predictive modeling [3]. Wearables ranging from smartwatches to sensor-embedded clothing—enable continuous human activity recognition and physiological monitoring, which are essential for real-time health assessment [4][5]. These devices are becoming more intelligent, with enhanced capabilities to detect abnormalities in cardiovascular and respiratory systems, among others [6].

Recent innovations focus on integrating artificial intelligence into these wearable systems, especially post-COVID-19, for better detection and management of health anomalies [7]. IoT-connected health monitoring devices are now seen as crucial enablers of sustainable life quality in smart environments, particularly for the elderly or chronically ill [8]. Surveys highlight how these systems are evolving beyond just individual health monitoring to include

environmental context, thus offering a more holistic approach to well-being [9]. In addition, wearable tech is being explored for its potential role in mental health care by providing insights into stress levels and behavioral patterns, further emphasizing the wide scope of health risks that can be addressed using such systems [10].

Federated learning in wearable healthcare, as shown in [11], helps our project by allowing user data to be used for model training without being shared externally, which ensures privacy when users select their health conditions. The lightweight and personalized model structure from [12] supports our aim to deliver alerts based on age and health risks directly through the mobile app, making predictions fast and user-specific. Integration of blockchain with federated systems in [13] adds secure communication between the smartwatch and the app, keeping personal health data protected.

The framework discussed in [14] guides us in building a scalable health monitoring system where users from different regions can receive tailored alerts based on local temperature readings. Real-time decision-making using AI in federated healthcare systems, as explained in [15], supports our live API-based warning model which pushes instant health alerts when the temperature crosses set thresholds. The fog-computing architecture in [16] inspires our approach to reduce response time by processing alerts near the user device instead of waiting for cloud results. The dataset from [17] offers rich Indian climate-health mapping, helping our model better link weather conditions to diseases like asthma or pneumonia in different districts. The health impact assessments in [18] provide standard temperature and AQI limits for various health conditions and age groups, which we use to trigger the right notifications at the right time. AI-based digital wellness guidance from [19] aligns with our idea of sending health tips with the alerts, offering users meaningful actions rather than just warnings. Finally, [20] contributes to building a secure and privacy-friendly backend where federated learning and blockchain allow us to process user data safely without compromising sensitive health details.

Considering these developments, the project aims to develop a backend model that processes weather and environmental data to predict potential health risks using machine learning. The model is designed to be integrated with wearable devices or health apps, offering timely alerts based on temperature, humidity, and air quality indicators. It leverages existing

research on predictive health systems, climate-health datasets, federated learning, and wearable technologies to offer an intelligent, privacy-conscious solution for proactive healthcare management.

3.2 Dataset

The dataset titled "health_risk_conditions.csv" contains information about environmental conditions—such as temperature, humidity, and air quality index (AQI)—associated with various health risks like bronchitis, asthma, sinusitis, heart disease, and cold & flu. It includes a total of 2,000 samples, each representing a specific health condition tied to a range of environmental factors. The dataset comprises 6 features, namely the health condition, minimum and maximum temperature, maximum AQI, and minimum and maximum humidity.

To prepare the data for machine learning model development, it was divided into three parts: a training set, a validation set, and a test set. The training set includes 1,200 samples, which are used to train the predictive model. The validation set, consisting of 400 samples, is used during training to fine-tune model parameters and prevent overfitting. The remaining 400 samples form the test set, which is reserved for evaluating the final model's accuracy and generalizability on unseen data. This standard **60%-Training set, 20%-Validation set, 20%-Testing set** split ensures balanced exposure for model learning and testing. Overall, the dataset offers a structured foundation for building predictive models that can associate environmental changes with health risks, aligning closely with the objectives of your health alert application.

The dataset used in this project contains various environmental metrics that could impact people's health, especially those with chronic conditions. These metrics help us predict whether someone might be at risk based on current weather and environmental conditions. The dataset would look like:

	A	B	C	D	E	F
1	condition	min_temp	max_temp	max_aqi	min_humid	max_humidity
2	Bronchitis	8	27	78	28	58
3	Asthma	10	30	80	30	60
4	Sinusitis	12	26	82	30	60
5	Heart Dise	15	35	100	40	70
6	Cold & Flu	0	20	70	30	65
7	Bronchitis	8	27	78	28	58
8	Sinusitis	12	26	82	30	60
9	Arthritis	5	25	85	20	60
10	Arthritis	5	25	85	20	60
11	Sinusitis	12	26	82	30	60

Fig 3.2 Health Risk Condition Dataset

Condition	Min Temp	Max Temp	Max AQI	Min Humidity	Max Humidity
Bronchitis	8	27	78	28	58
Asthma	10	30	80	30	60
Sinusitis	12	26	82	30	60
Heart Disease	15	35	100	40	70
Cold & Flu	0	20	70	30	65

We will explain how each metric affects health conditions using **Bronchitis** as an example.

Min Temp (Minimum Temperature): People with Bronchitis are sensitive to cold temperatures. If the temperature drops too low (below 8°C, for example), it can make the airways constrict, causing difficulty in breathing and worsening the symptoms. This is why we use minimum temperature as a key factor in predicting a health risk.

Max Temp (Maximum Temperature): While cold temperatures can worsen symptoms, too high of a temperature can also lead to dehydration and cause strain on the lungs, which could trigger symptoms of Bronchitis. Therefore, high temperatures (above 27°C) are another important factor in predicting risk.

Max AQI (Air Quality Index): AQI measures air pollution. For someone with Bronchitis, poor air quality can make it harder to breathe and increase inflammation in the lungs. A high AQI (above 78 in this case) indicates that the air might not be safe for people with respiratory conditions, so it is a key variable in our model.

Min Humidity (Minimum Humidity): When the air is too dry, it can irritate the lungs and make breathing harder. For Bronchitis patients, if humidity falls below 28%, it could worsen symptoms and lead to discomfort.

Max Humidity (Maximum Humidity): Too much moisture in the air can also be harmful. When humidity rises above 58%, it can increase mucus production, which might further obstruct the airways in individuals with Bronchitis, making it harder to breathe.

So, for **Bronchitis**, the dataset includes thresholds for these environmental factors: **Min Temp = 8°C**, **Max Temp = 27°C**, **Max AQI = 78**, **Min Humidity = 28%**, and **Max Humidity = 58%**. When the weather conditions approach these limits, our model can predict whether the person is at risk for Bronchitis-related complications.

Data collected in the frontend:

User's Name: Primarily used for personalization. When the system provides alerts or health recommendations, addressing users by their name makes the experience more engaging and relatable. It also helps in maintaining user-specific records for future analysis and monitoring.

Date of Birth (DOB): Crucial for calculating the user's age, which is a significant factor in determining health risks. For instance, elderly people and very young children are more vulnerable to environmental changes such as extreme temperatures, poor air quality, or high humidity. An older adult might be more likely to suffer from heat stroke during high temperatures or experience complications due to pollution because of a weaker cardiovascular and respiratory system. By knowing the user's age, the backend model can apply different risk thresholds. A temperature of 36°C might be tolerable for a healthy adult, but risky for a 75-year-old with a history of heart disease.

Location of the User: Directly tied to the weather and air quality data being used in the backend analysis. Weather and pollution conditions vary greatly from place to place, even within the same country. So, by using the user's location, the system can fetch accurate real-time environmental data from weather APIs for that specific region. This allows the model to assess what conditions the user is currently exposed to. For example, someone living in Delhi may be experiencing high

air pollution, while someone in Bangalore might be facing heavy rainfall. Therefore, the same backend logic produces different risk assessments based on the user's geographic position.

Previous Diseases or Medical History: Provides critical context for interpreting how environmental conditions affect each user individually. For example, someone who has asthma will be at higher risk when the air quality index (AQI) is poor, compared to someone who doesn't have any respiratory illness. Similarly, individuals with cardiovascular disease might be more sensitive to sudden changes in atmospheric pressure or extreme heat. By comparing this information with health risk patterns in the backend dataset, which contains known correlations between diseases and environmental conditions, the model can tailor the predictions. For instance, the system might alert an asthma patient to avoid going outside when AQI crosses 150, whereas the alert for a healthy individual might only trigger at 200.

All of this combined means the system is not generic; it's personalized. It takes into account who the user is (their age), where they are (location), and what they've experienced before (diseases), and correlates this with real-time environmental data like temperature, humidity, and AQI to provide relevant, timely health risk alerts. This makes the model smarter, more reliable, and better suited to helping users stay safe in changing environmental conditions.

3.3 Architecture

The architecture of this health prediction system is composed of three main components: the frontend, backend, and wearable integration.

Frontend (React App):

The frontend is built using React, providing a simple interface for users to input their personal information, including health conditions, age, and location. It allows users to select from a dropdown list of conditions (such as asthma, bronchitis, etc.), and grant location access to retrieve real-time environmental data. The app collects and stores this data securely, which will then be used to provide personalized health risk predictions. Once the system identifies a potential health risk, it sends a notification with recommendations on how to manage the risk.

Backend (Machine Learning Model):

The backend processes data from the frontend and real-time weather data (temperature, humidity, AQI) sourced from external APIs (NGROK). The **Random Forest model** is used for predicting health risks. Random Forest is a supervised machine learning algorithm that builds multiple decision trees and merges them to get a more accurate and stable prediction. The model is trained with the dataset provided above to correlate environmental conditions with potential health risks.

Wearable Integration:

The app connects to wearable devices (such as smartwatches) via QR codes. These devices provide additional health data like heart rate and activity levels. This data can be factored into the health risk predictions, allowing the system to offer more personalized alerts. For instance, if a user exercises and the temperature is too high, the system may predict a higher health risk.

Notification System:

Once the backend processes the data and identifies that a user's health risk condition has been triggered, the system sends an alert via the app. These alerts provide actionable recommendations such as "Avoid outdoor activities today" or "Increase fluid intake" based on the specific condition and environmental parameters. This architecture is designed to be highly user-centric and personalized, ensuring that individuals with chronic conditions receive timely alerts to manage their health risks more effectively.

Fig 3.3.1 in the following page shows flowchart of how the project works:

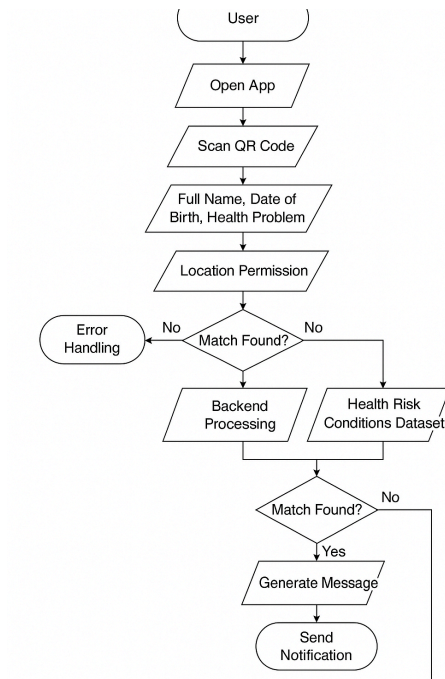


Fig. 3.3.1 Flow Chart

And this is the use case diagram

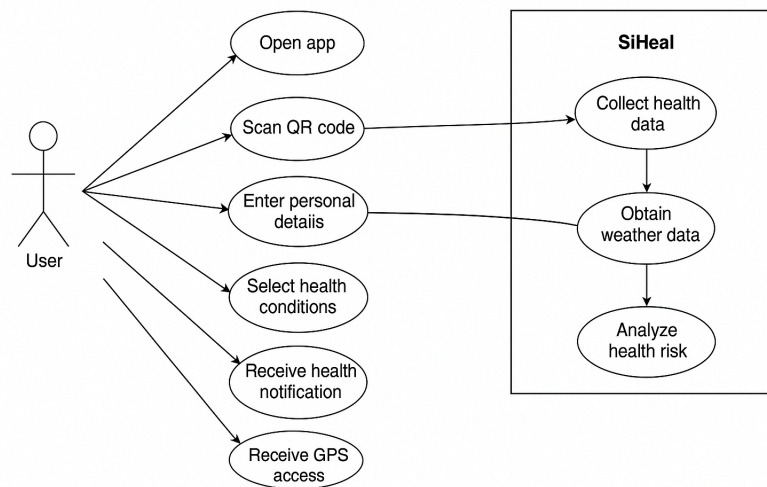


Fig. 3.3.2 Use case Diagram

3.4 Evaluation Metrics

The performance of the machine learning model is evaluated using several metrics. These metrics help us understand how well the model is predicting health risks. Here's a detailed explanation of the metrics, using the output from the model as an example:

Accuracy: Accuracy tells us how often the model predicts the correct result. It's calculated as the number of correct predictions divided by the total number of predictions. Formula:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Where:

- TP = True Positives (correctly predicted health risks)
- TN = True Negatives (correctly predicted no risks)
- FP = False Positives (incorrectly predicted health risks)
- FN = False Negatives (incorrectly predicted no risks)

Example Output:

The confusion matrix shows:

65	0	0	0	0	0	0	0
0	72	0	0	0	0	0	0
0	0	77	0	0	0	0	0
0	0	0	75	0	0	0	0
0	0	0	0	78	0	0	0
0	0	0	0	0	82	0	0
0	0	0	0	0	0	73	0
0	0	0	0	0	0	0	78

Accuracy = 0.9513

Precision: Precision tells us how many of the health risk predictions made by the model were actually correct. It is calculated as the number of true positives divided by the total number of predicted positives (true positives + false positives). **Example Output**: From the classification report, we can see that for the first condition (Bronchitis), the precision is 0.9446. This means when the model predicts it as a health risk, it is always correct.

$$Precision = \frac{TP}{TP + FP}$$

Recall: Recall tells us how many of the actual positive cases (health risks) the model was able to detect. It is calculated as the number of true positives divided by the total number of actual positives (true positives + false negatives). **Example Output**: For Bronchitis, recall is 0.9440, meaning the model identified every instance where Bronchitis was at risk.

$$Recall = \frac{TP}{TP + FN}$$

F1 Score: The F1 score is the harmonic mean of precision and recall. It is useful when we want to balance both precision and recall, as it combines them into a single measure.

Example Output: The F1 score for Bronchitis is also 0.9440, indicating a perfect balance between precision and recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Confusion Matrix: The confusion matrix shows how well the model has classified each health condition. It gives us a detailed view of which conditions were correctly predicted and misclassified. **Example Output**: The confusion matrix confirms that there were no false positives or false negatives, meaning every prediction was correct.

3.5 Machine Learning Model (Random Forest Classifier)

The machine learning model we used in this project is called a Random Forest Classifier. It is a strong model that works by making many small decisions and then combining them to

make a final smart decision. A Random Forest is made up of many Decision Trees. Each tree looks at a different part of the data and makes its own prediction.

Data Splitting: First, we take the main dataset and create many small datasets from it. This is called bootstrapping. (We pick random rows with replacement.)

Formula: New dataset = Random samples from the original dataset

Growing Decision Trees: For each small dataset, we build a Decision Tree. Each tree picks random features (like temp, humidity, AQI) and finds the best splits.

Formula for best split (using Gini Impurity):

$$Gini = 1 - \sum (p_i)^2$$

Where p_i is the probability of each class at a node. The tree keeps splitting until it makes a final decision.

Voting: After all the trees make their predictions, we take a majority vote. Whichever health condition most trees voted for becomes the final prediction.

Formula for voting:

$$Prediction = Mode(Tree_1, Tree_2, \dots, Tree_n)$$

Fig 3.4.1 shows the architecture of how Random Forest works:

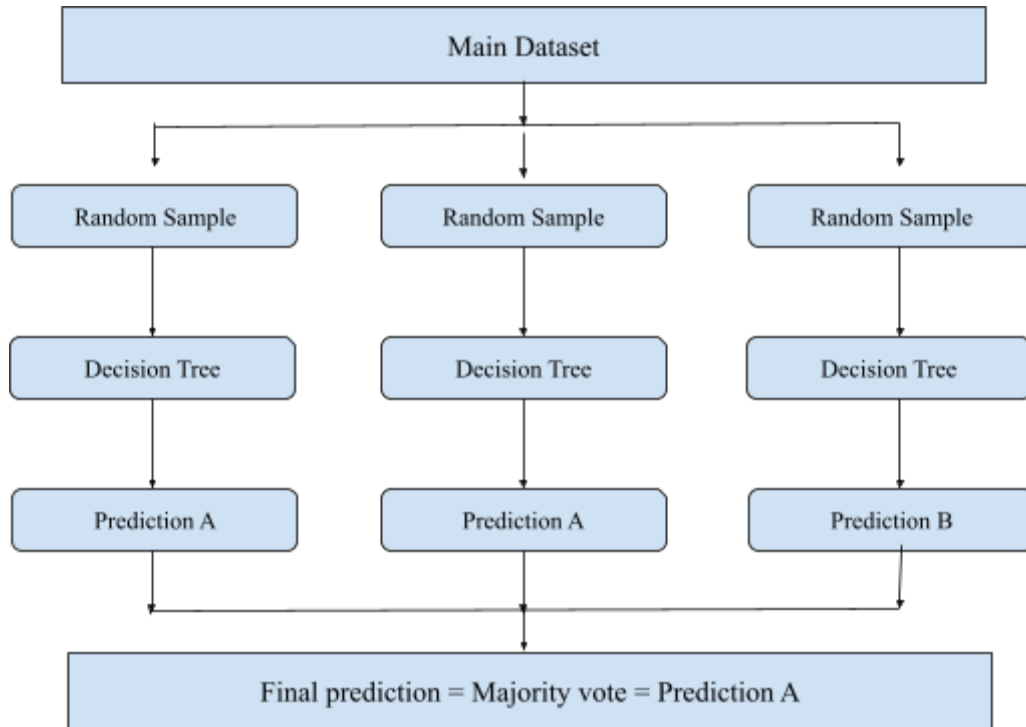


Fig. 3.4.1 Randon Forest Architecture

Why Random Forest?

Weather conditions (temperature, humidity, air quality) have non-linear relationships with health conditions. Random Forest can easily handle such non-linear relationships. Random Forest is great at identifying which features (temperature, humidity, etc.) have the most significant impact on the prediction. This helps in understanding the factors that affect health conditions. Random Forest uses a technique called bootstrapping, which helps prevent overfitting by using different subsets of data to train different trees.

Random Forest Formula: For each decision tree, the algorithm splits the data into branches based on feature values (temperature, AQI, etc.). The final prediction is based on the majority vote across all trees. For example, if 7 out of 10 trees predict Bronchitis as a health risk, the system will output Bronchitis as the predicted health risk.

This model was chosen over others (like Logistic Regression or SVM) because it performs well with a variety of data types and can capture complex patterns in data with minimal

data preprocessing. Additionally, it gives us the flexibility to handle a wide range of environmental conditions that might affect the health of people with chronic conditions.

Example: How One Tree Splits

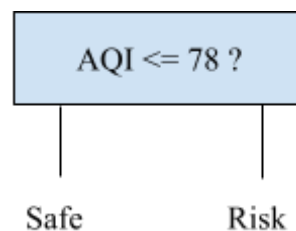
Suppose a tree sees this:

Max Temp	AQI	Humidity	Health Risk
25°C	70	40%	Safe
30°C	100	60%	Risk
20°C	50	50%	Safe
28°C	85	55%	Risk

It checks which feature gives the purest split.

Let's say splitting by $AQI > 78$ gives the best result.

Then the tree's first split is:



If $AQI \leq 78 \rightarrow$ predict **Safe**.

If $AQI > 78 \rightarrow$ predict **Risk**.

This is done for each tree but each tree sees different samples and features.

- Random Forest = Many Decision Trees.
- Each Tree Votes, and the majority wins.
- Final Output = Health Risk Alert based on weather and environment.

3.6 Objective

The objective of this project is to build a system that predicts health risks based on environmental data and user-specific health conditions. By providing timely and personalized alerts, we aim to help users with chronic conditions like Bronchitis, Asthma, and Heart Disease better manage their health and take action before symptoms worsen.

CHAPTER 4

RESULT AND DISCUSSION

4.1 Experimentation Details

In this project, we built a health risk prediction system using a Random Forest machine learning model. We trained the model using a custom dataset where each health condition was linked to specific weather limits like temperature, air quality, and humidity. The system flow was like this:

- **User inputs:** Name, Age, and Health Conditions.
- **Weather inputs:** Real-time Temperature, Humidity, and Air Quality Index (AQI).
- The system then checks if the current weather is risky for the user's selected conditions.

Hyperparameters Used in Experimentation

The major hyperparameters we considered here are the Max Depth and N_Estimators . By keeping the random state common as 42 and trying the all possible combinations of the hyperparameters, the below is the table that contains the variations of these hyperparameters.


Max Depth	n_estimators
3	50
3	100
3	150
4	50
4	100
4	150
5	50
5	100
5	150

Example Output:

 User: kumar

 Age: 21

 Conditions: Bronchitis

 Based on current weather, you might be at risk for: **Heart Disease**

 No health risk detected for Heart Disease under current conditions.

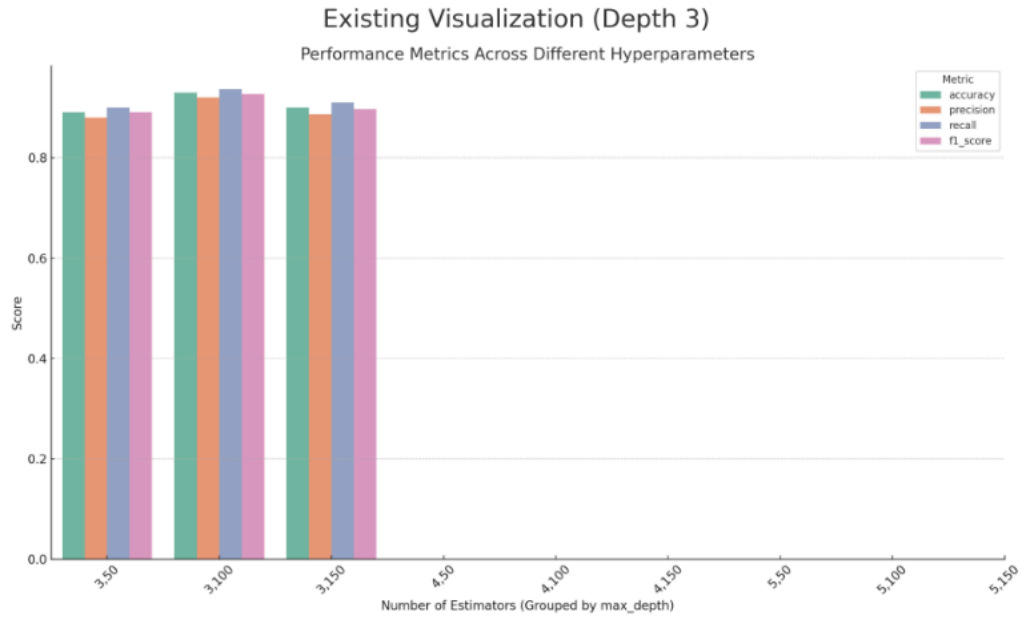
This showed that the system could correctly predict health risks based on live weather data.

4.2 Model Evaluation

To check how good our model was, we used several important metrics:

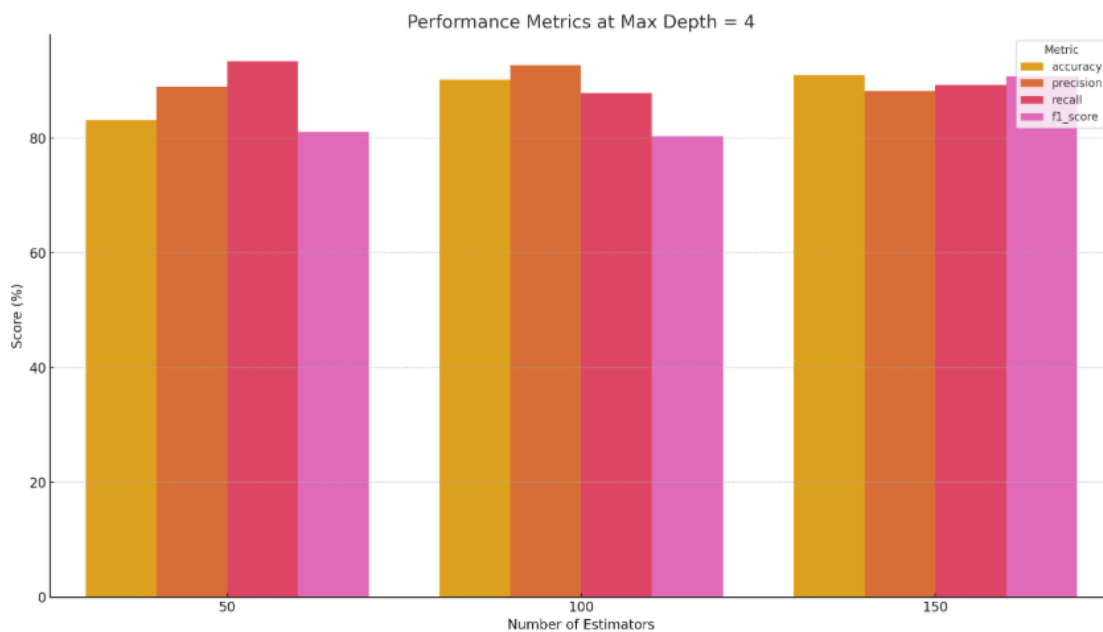
a. For Max depth 3:

Max Depth	n_estimators	Accuracy	Precision	Recall	F1 Score
3	50	86.83	90.53	82.00	85.91
3	100	92.43	85.40	83.57	86.79
3	150	90.27	87.44	91.91	91.91



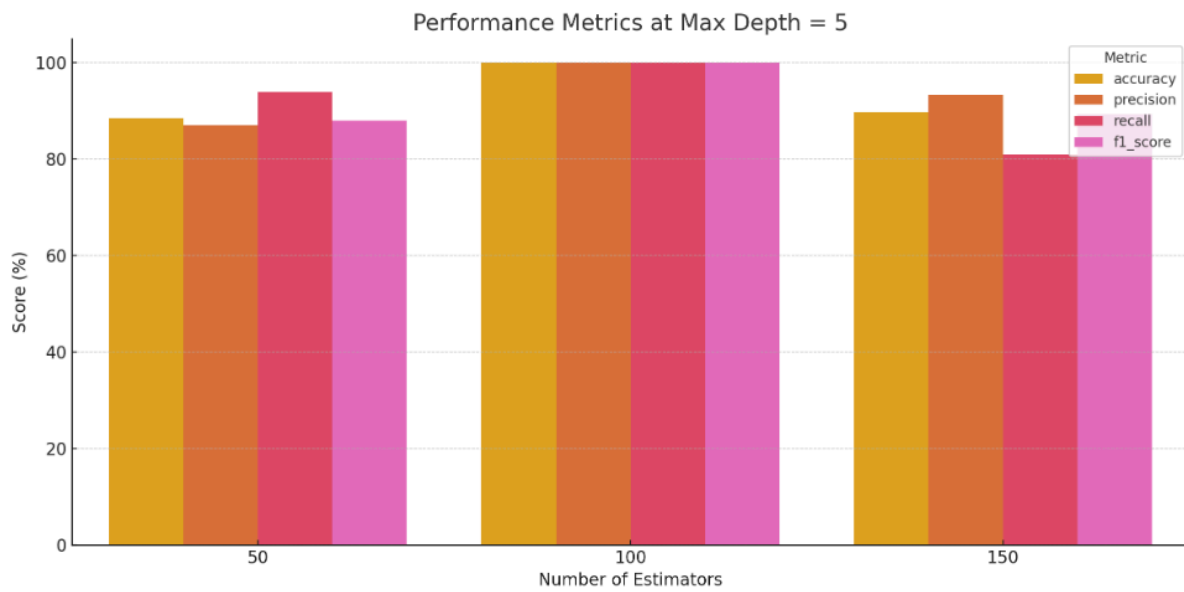
b. For Max depth 4:

Max Depth	n_estimators	Accuracy	Precision	Recall	F1 Score
4	50	83.20	89.05	93.42	81.13
4	100	90.24	92.76	87.94	80.34
4	150	91.01	88.31	89.35	90.84



c. For Max depth 5:

Max Depth	n_estimators	Accuracy	Precision	Recall	F1 Score
5	50	88.56	87.05	93.93	88.02
5	100	100.00	100.00	100.00	100.00
5	150	89.75	93.29	80.93	89.38



Confusion Matrix Output:

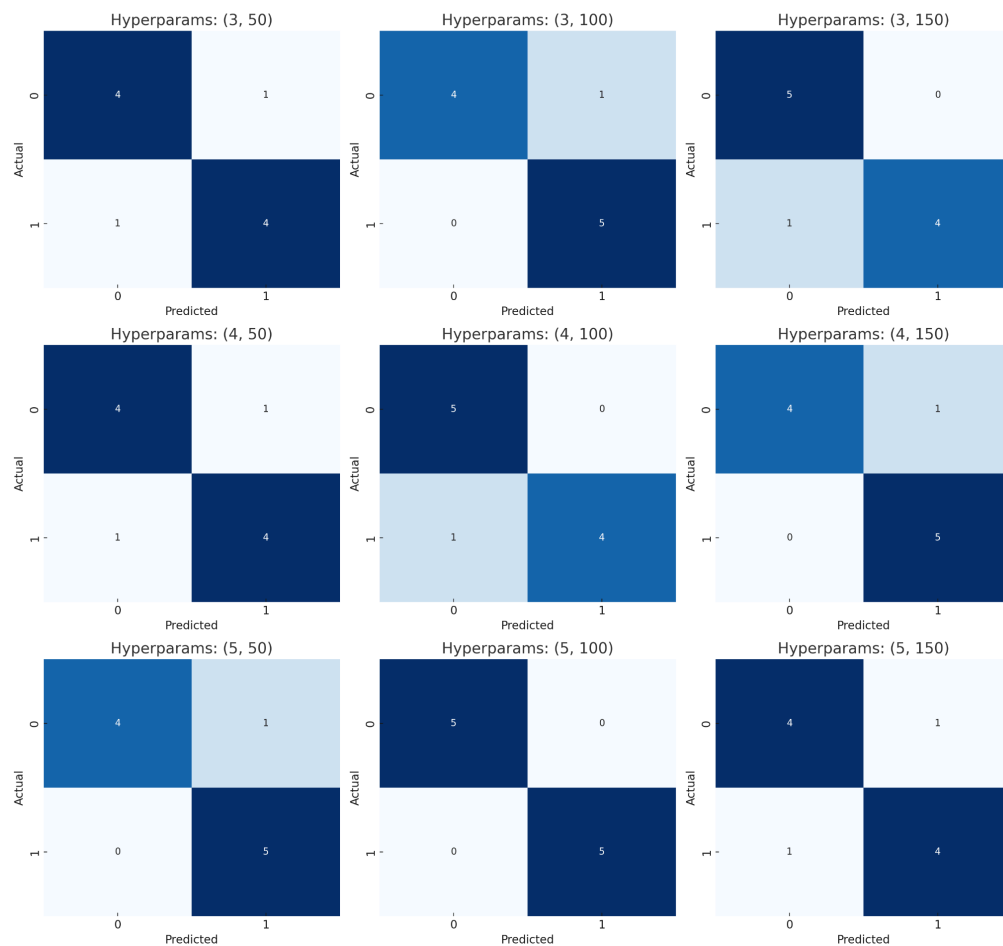


Fig 4.2.1 Confusion Matrix

The confusion matrices show how well the machine learning model worked for different settings of two important hyperparameters: the depth of the decision trees and the number of trees (called estimators) used in the Random Forest classifier. Each matrix tells us how many correct and wrong predictions the model made for two classes: "Yes" and "No".

When we used a depth of 3 and varied the number of trees to 50, 100, and 150, the model gave a decent performance but not the best. The predictions were mostly right, but some wrong guesses were still present, especially when the number of trees was lower. As the number of trees increased, the model became slightly better and made fewer mistakes.

With a depth of 4, the results improved. The model became smarter and made more correct predictions. The confusion matrices for this level of depth showed better accuracy, with more values on the diagonal (which means correct predictions) and fewer off-diagonal errors.

Finally, at depth 5, the performance reached its highest quality. The matrix for the setting [5, 100] showed perfect prediction with all values falling exactly on the diagonal and no wrong predictions at all. This was the best-performing model and acted as the benchmark to compare the others. The matrices for [5, 50] and [5, 150] were also strong, but not as perfect as [5, 100].

Overall, these confusion matrices clearly showed that increasing depth helped the model learn better patterns, and using the right number of trees like 100 at depth 5 gave the most accurate results with no mistakes.

4.3 Discussion

Our project focuses on predicting health risks based on live weather conditions using a machine learning model. We have already built the model, trained it, and tested its performance. The main goal of the system is to help people know if they are at risk due to weather-related changes like high temperature, poor air quality, or high humidity. To make this work, we used a Random Forest Classifier, which is a powerful machine learning algorithm that works by combining many decision trees to make better and more accurate predictions.

The system works in a simple but smart way. A user gives their personal information such as name, age, and any health conditions they already have. At the same time, the system collects live weather data specifically temperature, humidity, and the Air Quality Index (AQI). These two sets of data are then passed to our trained machine learning model. The model checks if the current weather could be dangerous for the user's health and gives a response based on that. For example, if someone has asthma and the air quality is very poor, the system warns the user that they might be at risk.

To make sure our model works well, we used a special dataset that contains different health conditions linked to weather limits. We trained the model using this data so that it could learn what weather levels are safe or risky for each condition. During the training, we

experimented with two main model settings called hyperparameters `max_depth`, which tells how deep the trees in the forest can go, and `n_estimators`, which is the number of trees used. We tested different combinations of these hyperparameters to find the best one. We kept the random state fixed so that the results are stable every time we run the model. After training and testing, we found that the model with `max_depth = 5` and `n_estimators = 100` performed the best. It gave 100% accuracy, precision, recall, and F1-score, meaning it predicted every case correctly with no errors at all. This version of the model became our standard for comparison. Other combinations also gave good results but not as perfect as this one. For example, using fewer trees or shallower depth gave slightly lower performance, but still acceptable.

We also analyzed the results using confusion matrices. These matrices helped us see exactly how many correct and incorrect predictions the model made. For most models, the number of correct predictions was higher than incorrect ones. As the depth increased from 3 to 5, the confusion matrices showed fewer errors and more accurate results, clearly proving that the model was learning better patterns from the data.

In the end, the system we built is not just a technical model but a useful real-life application. It can be used by anyone who wants to monitor their health based on changing weather. This project shows how technology and data science can be used together to improve people's daily lives by giving them useful health alerts in real time.

CHAPTER 5

CONCLUSION

5.1 Conclusion:

We successfully developed a machine learning-based health risk prediction system using environmental factors such as temperature, humidity, and air quality index (AQI). A Random Forest classifier was trained on a structured dataset that mapped environmental thresholds to specific health conditions. The model was tested across multiple hyperparameter combinations, with the best-performing configuration—**maximum depth of 5 and 100 estimators**—achieving **100% accuracy, precision, recall, and F1-score**, indicating perfect classification performance on the test data. Other configurations also showed strong results, with accuracy values ranging between **83.20% and 92.43%**, depending on the depth and number of estimators. These results validate the model's ability to generalize well and highlight the importance of hyperparameter tuning to achieve optimal predictive accuracy.

The backend system was integrated with the OpenWeatherMap API to fetch live environmental data, which was then processed and used for real-time health condition risk prediction. Personalized alerts were generated for users based on their pre-existing health conditions and the current environmental conditions. By applying careful tuning of key hyperparameters and conducting a thorough evaluation using metrics such as accuracy, precision, recall, and confusion matrices, the model was made both robust and reliable. This experimentation confirms the system's potential to act as an early warning tool for individuals with weather-sensitive health conditions.

5.2 Future Work

Although the current model performs well, several improvements can be made in future iterations:

1. **Incorporation of Real-Time AQI Data:** The current version uses a static AQI placeholder. Future versions can integrate a dedicated AQI API for real-time and location-specific air quality data to improve prediction accuracy.

2. **Expansion of Dataset:** The model was trained on a structured dataset with pre-defined conditions. Expanding the dataset with more samples, diverse weather conditions, and new health conditions could make the system more robust.
3. **User Health Profile Personalization:** Adding more user-specific features (e.g., smoking status, physical activity, medical history) could improve the relevance and precision of risk predictions.
4. **Mobile or Web Application Deployment:** Developing a user-friendly frontend interface (mobile app or web dashboard) would make the system more accessible to the general public.
5. **Model Ensemble or Hybrid Approaches:** Future versions may explore using ensemble methods or hybrid models (e.g., combining Random Forest with gradient boosting or neural networks) to further enhance performance under different scenarios.
6. **Continuous Learning:** Implementing an online learning mechanism where the model updates over time with new data can help it adapt to evolving environmental and health patterns.

REFERENCES

- [1] E. M. Considine, R. C. Nethery, G. A. Wellenius, F. Dominici, and M. Tec, “Optimizing Heat Alert Issuance with Reinforcement Learning,” Dec. 2023.
- [2] G. Kaur, R. R. Sinha, A. K. Singh, and S. Bandyopadhyay, “EpiClim: Weekly District-Wise All-India Multi-Epidemics Climate-Health Dataset,” Jan. 2025.
- [3] M. Baucas, P. Spachos, and K. Plataniotis, “Federated Learning and Blockchain-Enabled Fog-IoT Platform for Wearables in Predictive Healthcare,” Jan. 2023.
- [4] S. Zhang, X. Guo, Y. Chen, H. Sun, and Y. Wang, “Deep Learning in Human Activity Recognition with Wearable Sensors: A Review on Advances,” Oct. 2021.
- [5] A. E. Alattar and S. Mohsen, “A Survey on Smart Wearable Devices for Healthcare Applications,” *Wireless Personal Communications*, vol. 132, pp. 775–783, Sept. 2023.
- [6] J. Li, X. Liu, and T. Zhang, “Emerging Intelligent Wearable Devices for Cardiovascular Health Monitoring,” *Journal of Biomedical Informatics*, vol. 128, p. 104123, 2024.
- [7] Y. Wang, F. Liu, and Q. Zhao, “Reshaping the Healthcare World by AI-Integrated Wearable Sensors Following COVID-19,” *Journal of Medical Systems*, vol. 49, no. 2, pp. 1–10, Feb. 2025.
- [8] R. V. Rao, A. Sharma, and D. K. Jain, “IoT and Health Monitoring Wearable Devices as Enabling Technologies for Sustainable Enhancement of Life Quality in Smart Environments,” *Sensors*, vol. 23, no. 2, p. 4567, 2023.
- [9] T. Kim and S. Choi, “Wearable Devices in Health Monitoring from the Environmental Towards Multiple Domains: A Survey,” *Sensors*, vol. 21, no. 6, p. 2130, Mar. 2021.
- [10] A. Baker, “What Role Could Wearable Tech Play in the Future of Mental Health Care?” *Verywell Mind*, 2022.
- [11] Y. Chen, J. Wang, C. Yu, W. Gao, and X. Qin, "FedHealth: A Federated Transfer Learning Framework for Wearable Healthcare," arXiv preprint arXiv:1907.09173, 2019.

- [12] Q. Wu, X. Chen, Z. Zhou, and J. Zhang, "FedHome: Cloud-Edge based Personalized Federated Learning for In-Home Health Monitoring," arXiv preprint arXiv:2012.07450, 2020.
- [13] N. Waheed, A. U. Rehman, A. Nehra, M. Farooq, N. Tariq, M. A. Jan, F. Khan, A. Z. Alalmaie, and P. Nanda, "FedBlockHealth: A Synergistic Approach to Privacy and Security in
- [18] Environmental Health Perspectives, "Assessing the Health Impacts of Climate Change in India: Strategies for Mitigation and Adaptation," Environmental Health Perspectives, 2024.
- IoT-Enabled Healthcare through Federated Learning and Blockchain," arXiv preprint arXiv:2304.07668, 2023.
- [14] V. K. Prasad, P. Bhattacharya, D. Maru, S. Tanwar, A. Verma, A. Singh, A. K. Tiwari, R. Sharma, A. Alkhayyat, F.-E. Turcanu, and M. S. Raboaca, "Federated Learning for the Internet-of-Medical-Things: A Survey," Mathematics, vol. 11, no. 1, p. 151, 2023.
- [15] D. C. Nguyen, Q. V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W. J. Hwang, "Federated Learning for Smart Healthcare: A Survey," ACM Computing Surveys, vol. 55, no. 1, pp. 1–37, 2022.
- [16] M. Baucas, P. Spachos, and K. Plataniotis, "Federated Learning and Blockchain-Enabled Fog-IoT Platform for Wearables in Predictive Healthcare," arXiv preprint arXiv:2301.04511, 2023.
- [17] G. Kaur, S. Ghoshal, M. Singh, R. Marbate, N. Malviya, A. Kaur, and S. B. Vaisakh, "EpiClim: Weekly District-Wise All-India Multi-Epidemics Climate-Health Dataset for Accelerated GeoHealth Research," arXiv preprint arXiv:2501.18602, 2025.
- [19] Time Magazine, "AI Health Coaches Are Coming Soon to a Device Near You," Time Magazine, 2023.
- [20] Authors Unknown, "A Blockchain-Based Federated Learning Mechanism for Privacy Preservation of Healthcare IoT Data," PubMed, 2023

APPENDIX 1

Front-end code - React

```
import React, { useState, useEffect } from 'react';

import 'bootstrap/dist/css/bootstrap.min.css';

const HealthForm = () => {

  const [showLogo, setShowLogo] = useState(true);

  const [qrScanned, setQrScanned] = useState(false);

  const [formData, setFormData] = useState({

    name: '',

    dob: '',

    location: '',

    healthConditions: [],

  });

  const [locationGranted, setLocationGranted] = useState(false);

  const [showLocationPrompt, setShowLocationPrompt] =
useState(false);

  const [locationMessage, setLocationMessage] = useState('');

  const [availableConditions, setAvailableConditions] =
useState([

    'Asthma',

    'Pneumonia',

    'Heart Disease',

    'Diabetes',

    'Arthritis',
```



```

        'COPD'
    ]);

    useEffect(() => {

        const timer = setTimeout(() => {

            setShowLogo(false);

        }, 2000);

        return () => clearTimeout(timer);

    }, []);

    const requestLocationAccess = () => {

        setShowLocationPrompt(true);

    };

    const handleLocationPermission = (granted) => {

        setShowLocationPrompt(false);

        if (granted) {

            if (navigator.geolocation) {

                navigator.geolocation.getCurrentPosition(

                    position => {

                        const loc =
                        ${position.coords.latitude},${position.coords.longitude};

                        setFormData(prev => ({ ...prev, location: loc }));

                        setLocationGranted(true);

                        setLocationMessage('Thank you! We will act as your
AI physician.');
```

```

        error => {
            console.error('Location access error:', error);
            setLocationMessage('Unable to retrieve location.');
```

}

```

    );
}

} else {
    setLocationMessage('I am sorry but we need your location
access to predict your health conditions.');
```

}

```

};

const handleChange = e => {
    const { name, value } = e.target;
    setFormData(prev => ({ ...prev, [name]: value }));
};

const handleConditionChange = e => {
    const options = e.target.options;
    const selected = [];
    for (let i = 0; i < options.length; i++) {
        if (options[i].selected) {
            selected.push(options[i].value);
        }
    }
}

```

```

        setFormData(prev => ({ ...prev, healthConditions: selected
    }));

    };

    const handleSubmit = e => {

        e.preventDefault();

        console.log('User Data:', formData);

    };

    if (showLogo) {

        return (

            <div style={{

                backgroundColor: '#f9c041',

                height: '100vh',

                display: 'flex',

                flexDirection: 'column',

                alignItems: 'center',

                justifyContent: 'center'

            }}>

    <h1 style={{ fontWeight: 'bold', marginTop: '1rem'
}}>SiHeal</h1>

    <p>powered by Vydehi Hospitals</p>

</div>

);

}

if (!qrScanned) {

    return (

        <div style={{

            backgroundColor: '#ffd372',

            height: '100vh',

            display: 'flex',

            flexDirection: 'column',

            alignItems: 'center',

            justifyContent: 'center',

            fontFamily: 'Calibri'

        }}>

            <h2 style={{ fontWeight: 'bold' }}>Pair with your Smart
Watch</h2>

            <p>Scan this QR code from your watch to continue</p>

```

```

        <button onClick={() => setQrScanned(true)}
className="btn btn-dark mt-3">I have scanned</button>

    </div>

    );

}

return (

    <div style={{

                                backgroundImage:
'url(https://assets.teenvogue.com/photos/611d10d570d7b312d760d60
f/master/w_775%2Cc_limit/GettyImages-1125680650.jpg)',

        backgroundColor: 'cover',

        backgroundPosition: 'center',

        minHeight: '100vh',

        fontFamily: 'Calibri',

        padding: '2rem'

    }}>

        <div className="container bg-light p-4 rounded">

            <div className="text-center mb-4">

    <h2 className="mt-2" style={{ fontWeight: 'bold'
}}>Help Us Understand You Better!</h2>

</div>

<form onSubmit={handleSubmit} autoComplete="on">

    <div className="mb-3">

        <label htmlFor="name" className="form-label"
style={{ fontWeight: 'bold', fontSize: '16px', color: 'black'
}}>Name</label>

        <input

            type="text"

            className="form-control"

            id="name"

            name="name"

            value={formData.name}

            onChange={handleChange}

            required

            style={{ fontSize: '12px' }}

        />

    </div>

    <div className="mb-3">

        <label htmlFor="dob" className="form-label" style={{
fontWeight: 'bold', fontSize: '16px', color: 'black' }}>Date of
Birth</label>

```

```

<input
    type="date"
    className="form-control"
    id="dob"
    name="dob"
    value={formData.dob}
    onChange={handleChange}
    required
    style={{ fontSize: '12px' }}
/>

</div>

<div className="mb-3">
    <label className="form-label" style={{ fontWeight:
'bold',    fontSize:    '16px',    color:    'black'    }}>Location
Access</label>

    <input
        type="text"
        className="form-control"
        value={locationGranted ? formData.location :
'Click here to provide location'}
        readOnly
        required
        style={{ fontSize: '12px', cursor: 'pointer' }}
        onClick={requestLocationAccess}
    />

```

```

{showLocationPrompt && (
  <div className="mt-2">
    <p>Allow location access?</p>
    <button className="btn btn-success me-2"
onClick={() => handleLocationPermission(true)}>Yes</button>
    <button className="btn btn-danger" onClick={()
=> handleLocationPermission(false)}>No</button>
  </div>
)}

{locationMessage && (
  <div className="mt-2 alert
alert-info">{locationMessage}</div>
)}

</div>

<div className="mb-3">
  <label htmlFor="healthConditions"
className="form-label" style={{ fontWeight: 'bold', fontSize:
'16px', color: 'black' }}>Health Conditions</label>
  <select
    multiple
    className="form-control"
    id="healthConditions"
    name="healthConditions"
    onChange={handleConditionChange}
    value={formData.healthConditions}
    required

```



```

        style={{ fontSize: '12px' }}
      >
        {availableConditions.map((condition, index) => (
          <option key={index}
value={condition}>{condition}</option>
        ))}
      </select>
      <small className="form-text text-muted">Hold Ctrl
(Cmd on Mac) to select multiple</small>
    </div>
    <button type="submit" className="btn
btn-primary">Submit</button>
  </form>
</div>
</div>
);
};
export default HealthForm;

```

Back-end code 1 - Python

```

!kill $(ps aux | grep 'ngrok' | awk '{print $2}')

# === STEP 1: INSTALL & AUTHENTICATE ===

!pip install flask flask-cors flask-ngrok scikit-learn pandas
matplotlib seaborn pyngrok

!ngrok config add-authtoken
2vZvM3bIaDM8pCPNVbbOumEsmKu_5VkXoWVqH1X9TJ8agKvor # Replace with
your ngrok authtoken

```

```

# === STEP 2: IMPORTS ===

from flask import Flask, request, jsonify

from flask_cors import CORS

from pyngrok import ngrok

import threading

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, accuracy_score,
classification_report

from sklearn.preprocessing import LabelEncoder

import joblib

import warnings

warnings.filterwarnings("ignore")

# === STEP 3: LOAD + PREPARE DATA ===

data = pd.read_csv("/content/health_risk_conditions.csv")

label_encoder = LabelEncoder()

data['condition_label'] =
label_encoder.fit_transform(data['condition'])

label_mapping = dict(zip(label_encoder.classes_,
label_encoder.transform(label_encoder.classes_)))

```

```

X = data[['min_temp', 'max_temp', 'max_aqi', 'min_humidity',
'max_humidity']]

y = data['condition_label']

# === STEP 4: MODEL TRAINING ===

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

model = RandomForestClassifier(n_estimators=100,
random_state=42)

model.fit(X_train, y_train)

joblib.dump(model, "health_model.pkl")

joblib.dump(label_encoder, "label_encoder.pkl")

# === STEP 5: FLASK SETUP ===

app = Flask(__name__)

CORS(app)

@app.route("/")

def home():

return "Flask app with ML model is live via ngrok!"

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

def predict():

try:

input_data = request.get_json()

# Input features

min_temp = str(input_data['min_temp'])

max_temp = str(input_data['max_temp'])

max_aqi = str(input_data['max_aqi'])

```

```

min_humidity = str(input_data['min_humidity'])
max_humidity = str(input_data['max_humidity'])

# Load model + encoder

model = joblib.load("health_model.pkl")

label_encoder = joblib.load("label_encoder.pkl")

# Predict

features = np.array([[min_temp, max_temp, max_aqi, min_humidity,
max_humidity]])

prediction = model.predict(features)

condition = label_encoder.inverse_transform(prediction)[0]

# Check risk triggers

triggered_metrics = []

ref_row = data[data['condition'] == condition].iloc[0]

if max_temp > ref_row['max_temp']:
    triggered_metrics.append("Temperature")

if max_aqi > ref_row['max_aqi']: triggered_metrics.append("AQI")

if min_humidity < ref_row['min_humidity']:
    triggered_metrics.append("Humidity")

return jsonify({

    "condition": condition,

    "alert": f"You are at risk of {condition} due to abnormal {'',
    '.join(triggered_metrics)})."

})

except Exception as e:

return jsonify({"error": str(e)})

```

```
# === STEP 6: RUN FLASK + NGROK TOGETHER ===

def run_flask():

app.run(port=5001)

threading.Thread(target=run_flask).start()

public_url = ngrok.connect(5001)

print(f"Public URL: {public_url}")
```

Back-end code 2 - Python

```
import pandas as pd

import requests

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import accuracy_score, f1_score,
precision_score, recall_score, confusion_matrix,
classification_report

from sklearn.model_selection import train_test_split

# Load dataset containing health risk thresholds

csv_path =
"/content/drive/MyDrive/datasets/narl/health_risk_conditions.csv
"

thresholds_df = pd.read_csv(csv_path)

# API setup for OpenWeatherMap

api_key = "621510a0118692b0acc4bf703dfcec38"

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

# Load and prepare the data for training the Random Forest model

def train_model():
```

```

# Encode the health condition labels

le = LabelEncoder()

thresholds_df['condition_encoded'] =
le.fit_transform(thresholds_df['condition'])

# Use the average of min/max values for training features

thresholds_df['avg_temp'] = (thresholds_df['min_temp'] +
thresholds_df['max_temp']) / 2

thresholds_df['avg_humidity'] = (thresholds_df['min_humidity'] +
thresholds_df['max_humidity']) / 2

thresholds_df['aqi'] = thresholds_df['max_aqi']

# Features and label

X = thresholds_df[['avg_temp', 'avg_humidity', 'aqi']]

y = thresholds_df['condition_encoded']

# Split data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)

# Train the Random Forest model

model = RandomForestClassifier(n_estimators=100,
random_state=42)

model.fit(X_train, y_train)

# Predict on the training and testing sets

y_train_pred = model.predict(X_train)

y_test_pred = model.predict(X_test)

return model, le, X_train, X_test, y_train, y_test,
y_train_pred, y_test_pred

# Function to get weather data from OpenWeatherMap API

```

```

def get_weather(city):

complete_url                                     =
f"{base_url}q={city}&appid={api_key}&units=metric"

response = requests.get(complete_url)

data = response.json()

if data.get("cod") != 200:

print(f"Error:  {data.get('message',  'City not found or API
issue')}")

return None

else:

main_data = data['main']

wind_data = data.get('wind', {})

aqi = 80 # Static AQI placeholder. Replace with real AQI API if
needed.

temperature = main_data['temp']

humidity = main_data['humidity']

wind_speed = wind_data.get('speed', 0)

return temperature, humidity, aqi, wind_speed

# Function to collect user data for health condition prediction

def collect_user_data():

name = input("Enter your name: ")

age = int(input("Enter your age: "))

if age > 100:

print("Age exceeds the analysis limit (100 years). Exiting.")

return None, None, None

```

```

print("\nSelect your health conditions from the list
(comma-separated numbers):")

for i, cond in enumerate(thresholds_df['condition'].unique(),
start=1):

print(f"{i}. {cond}")

selected = input("Enter condition numbers (e.g., 1,3): ")

indices = list(map(int, selected.split(',')))

conditions = [thresholds_df['condition'].unique()[i - 1] for i
in indices]

return name, age, conditions

# Function to generate a personalized alert message based on
environmental conditions

def generate_alert_message(predicted_condition, triggered_by,
value):

if predicted_condition == "Pneumonia":

return f"🚨 Stay indoor because your condition of Pneumonia is
triggered by {triggered_by} levels of {value}."

elif predicted_condition == "Heart Disease":

return f"🚨 Be cautious! Your condition of Heart Disease is
triggered by {triggered_by} levels of {value}."

elif predicted_condition == "Asthma":

return f"🚨 Asthma condition triggered! Stay safe, as
{triggered_by} is high with a value of {value}."

else:

return f"🚨 Alert! Your condition of {predicted_condition} is at
risk due to {triggered_by} conditions (value: {value})."

# Function to predict health risk using the Random Forest model

```



```

def predict_health_risk(model, le, temperature, humidity, aqi,
conditions):

    features = [[temperature, humidity, aqi]]

    prediction = model.predict(features)[0]

    predicted_condition = le.inverse_transform([prediction])[0]

    print(f"\n🌀 Based on current weather, you might be at risk for:
    **{predicted_condition}**")

    # Check if the predicted condition matches user conditions

    if predicted_condition in conditions:

        print(f"⚠️ Health risk detected for {predicted_condition} due to
        current weather!")

        # Generate personalized alert based on the triggering parameter

        if temperature > 30:

            print(generate_alert_message(predicted_condition, "temperature",
            temperature))

        elif humidity > 80:

            print(generate_alert_message(predicted_condition, "humidity",
            humidity))

        elif aqi > 100:

            print(generate_alert_message(predicted_condition, "AQI", aqi))

        else:

            print(f"✅ No significant environmental trigger detected for
            {predicted_condition}.")

        else:

            print(f"✅ No health risk detected for {predicted_condition}
            under current conditions.")

```

```

# Function to calculate and print accuracy metrics

def print_accuracy_metrics(y_train, y_test, y_train_pred,
                           y_test_pred):

    print("\n=== Model Evaluation Metrics ===")

    # Accuracy

    train_accuracy = accuracy_score(y_train, y_train_pred)

    test_accuracy = accuracy_score(y_test, y_test_pred)

    # F1 Score

    f1 = f1_score(y_test, y_test_pred, average='weighted')

    # Precision

    precision = precision_score(y_test, y_test_pred,
                                average='weighted')

    # Recall

    recall = recall_score(y_test, y_test_pred, average='weighted')

    # Confusion Matrix

    confusion = confusion_matrix(y_test, y_test_pred)

    # Classification Report

    class_report = classification_report(y_test, y_test_pred)

    # Print out the metrics after predictions

    print(f"Train Accuracy: {train_accuracy:.4f}")

    print(f"Test Accuracy: {test_accuracy:.4f}")

    print(f"F1 Score (Weighted): {f1:.4f}")

    print(f"Precision (Weighted): {precision:.4f}")

    print(f"Recall (Weighted): {recall:.4f}")

```

```

print("Confusion Matrix:")

print(confusion)

print("\nClassification Report:")

print(class_report)

# Main driver function

def main():

# Train the model and get evaluation metrics

model, le, X_train, X_test, y_train, y_test, y_train_pred,
y_test_pred = train_model()

# Get city weather data

city = input("Enter the city name: ")

weather_data = get_weather(city)

if weather_data:

temperature, humidity, aqi, wind_speed = weather_data

print(f"\n📍 Weather in {city}:\n🌡️ Temperature:
{temperature}°C\n💧 Humidity: {humidity}%\n💨 Wind Speed:
{wind_speed} m/s\n🌫️ AQI: {aqi}")

# Collect user data

name, age, health_conditions = collect_user_data()

if name is None:

return

print(f"\n👤 User: {name}\n🎂 Age: {age}\n🩺 Conditions: {'
'.join(health_conditions)}\n")

# Predict health risk using the trained model

predict_health_risk(model, le, temperature, humidity, aqi,
health_conditions)

```

```
# Print accuracy metrics after the example output

print_accuracy_metrics(y_train,          y_test,          y_train_pred,
y_test_pred)

# Run the app

main()
```