**Water Quality Prediction**

**Web App**

## PROJECT WORK PHASE 1 (REVIEW2)

***Submitted by***

# SHYAM KUMAR A 212221230098

***in partial fulfilment for the award of the degree of***

# BACHELOR OF TECHNOLOGY

***in***

**ARTIFICIAL INTELLIGENCE AND DATA SCIENCE**



**SAVEETHA ENGINEERING COLLEGE, THANDALAM**

**An Autonomous Institution Affiliated to**

# ANNA UNIVERSITY - CHENNAI 600 025

NOVEMBER 2024

****

ANNA UNIVERSITY, CHENNAI

# BONAFIDE CERTIFICATE

Certified that this Project report **“WATER QUALITY PREDICTION WEB APP”** is the bonafide work of **SHYAM KUMAR A (212221230098),** who carried out this project work under my supervision.

|  |  |
| --- | --- |
| SIGNATURE **Dr. Lavnya G**  **Associate Professor**  SUPERVISOR Dept of Artificial Intelligence and Data Science  Saveetha Engineering College, Chennai 602105 | SIGNATURE **Dr. Karthi Govindharaju, M.E., Ph.D.,**  **Professor** HEAD OF THE DEPARTMENT Dept of Artificial Intelligence and Data Science  Saveetha Engineering College, Chennai 602105. |

DATE OF THE VIVA VOCE EXAMINATION: …………………………

**ACKNOWLEDGEMENT**

I would like to express my heartfelt gratitude to our esteemed Founder President **Dr. N. M. Veeraiyan**, our President **Dr. Saveetha Rajesh**, our Director **Dr. S. Rajesh**, and the entire management team for providing the essential infrastructure.

I extend my sincere appreciation to our principal, **Dr. V. Vijaya Chamundeeswari, M.Tech., Ph.D.,** for creating a supportive learning environment for this project.

I am very thankful to our Dean of ICT, **Mr. Obed Otto,** M.E., for facilitating a conducive atmosphere that allowed me to complete my project successfully.

My thanks go to **Dr. Karthi Govindharaju, M.E., Ph.D.,**, Professor and Head of the Department of Artificial Intelligence and Data Science at Saveetha Engineering College, for his generous support and for providing the necessary resources for my project work.

I would also like to express my profound gratitude to my Supervisor, **Dr Lavanya G,** Associate Professor at Saveetha Engineering College, and my Project Coordinator **Dr. N.S. Gowri Ganesh**, Associate Professor at Saveetha Engineering College, for their invaluable guidance, suggestions, and constant encouragement, which were instrumental in the successful completion of this project. Their timely support and insights during the review process were greatly appreciated.

I am grateful to all my college faculty, staff, and technicians for their cooperation throughout the project. Finally, I wish to acknowledge my loving parents, friends, and well-wishers for their encouragement in helping me achieve this milestone

**ABSTRACT**

Water pollution is a pressing global issue, significantly impacting public health and ecosystems. Accurate prediction of water quality is crucial for effective monitoring, prevention, and remediation efforts. This project aims to develop a web application that employs machine learning algorithms to predict water quality based on various parameters such as pH, temperature, dissolved oxygen, turbidity, and conductivity. The application will provide a user-friendly interface for data input and visualization of predicted water quality.

The developed web application will utilize a combination of data collection, preprocessing, feature engineering, model selection, training, evaluation, and web development phases. By gathering historical water quality data from reliable sources and preprocessing it to ensure consistency and completeness, the application will lay a solid foundation for accurate predictions. Machine learning algorithms, such as linear regression, decision trees, or neural networks, will be employed to analyse the pre-processed data and establish relationships between water quality parameters and predicted values. The trained models will then be integrated into a user-friendly web interface, allowing users to input water quality data and receive real-time predictions.

The development of a water quality prediction web app presents a significant opportunity to address the pressing challenges of water pollution and ensure the sustainable management of water resources. By combining advanced machine learning techniques, user-friendly interfaces, and integration with IoT technologies, this project can provide a valuable tool for policymakers, researchers, and the public to monitor, understand, and protect water quality.

The successful implementation of such a web app can contribute to a healthier environment, improved public health, and a more sustainable future. The expected outcomes of this project include providing a valuable tool for real-time water quality monitoring, early warning systems, informed decision-making by water management authorities, and public awareness regarding water quality issues. By leveraging machine learning and web development technologies, this project aims to contribute to sustainable water resource management and environmental protection.

# TABLE OF CONTENTS

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CHAPTER NO.** | | | **TITLE** | **Page**  **Number** |
| **1** |  |  | **INTRODUCTION** |  |
|  | 1.1 | Overview of the project | 1 |
|  | 1.2 | Problem Definition | 2 |
| **2** |  |  | **LITERATURE SURVEY** | 3 |
| **3** |  |  | **SYSTEM ANALYSIS** |  |
|  | 3.1 |  | Existing System | 9 |
|  | 3.2  3.3  3.4 |  | Disadvantages Proposed System Advantages | 9  9  10  10 |
|  | 3.5 |  | Feasibility Study | 10 |
|  | 3.6 |  | Hardware Environment | 10 |
|  | 3.7 |  | Software Environment | 10 |
|  | 3.8 |  | Technologies Used | 10 |
|  |  | 3.8.1 | Python | 11 |
|  |  | 3.8.2 | Machine Learning | 11 |
| **4** |  |  | **SYSTEM DESIGN** |  |
|  | 4.1 |  | ER- Diagram | 12 |
|  | 4.2 |  | Data Flow Diagram | 13 |
|  | 4.3 |  | UML Diagram | 14 |
|  |  | 4.3.1 | Use Case Diagram | 14 |
|  |  | 4.3.2 | Class Diagram | 15 |
|  |  | 4.3.3 | Sequence Diagram | 16 |
| **5** |  |  | **SYSTEM ARCHITECTURE** |  |
|  | 5.1 |  | Architecture Diagram | 17 |
|  | 5.2 |  | Algorithms | 18 |
|  |  | 5.2.1 | XGB Classifier | 18 |
| **6** |  |  | **SYSTEM IMPLEMENTATION** |  |
|  | 6.1 | Module-1 | 20 |
|  |  | Data Collection and Preprocessing |  |
|  | 6.2 | Module-2 | 21 |
|  |  | Model Training |  |
|  | 6.3 | Module-3 | 22 |
|  |  | Prediction of Output |  |
| **7** |  |  | **SYSTEM TESTING** |  |
|  | 7.1  7.2 | Black Box Testing White Box Testing | 23  23  24 |
|  | 7.3 | Test Cases |  |
| **8** |  |  | **CONCLUSION AND FUTURE** |  |
|  |  | **-ENHANCEMENT** |  |
|  | 8.1 | Conclusion | 26 |
|  | 8.2 | Future Enhancement | 27 |
| **9** | 9.1 |  | **APPENDIX-1**  Source Code | 28 |
| **10** |  |  | **APPENDIX-2** |  |
|  |  | Sample Output |  |
|  | 10.1 | Home Page | 35 |
|  |  |  | 36 |
| **11** |  |  | **REFERENCES** | 39 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **XG Boost**  **XGB Classifier**  **WQPWA** | Extreme Gradient Boosting  Extreme Gradient Boosting Classifier  Water Quality Prediction Web App |

# Chapter 1

**INTRODUCTION**

## OVERVIEW OF THE PROJECT

Water quality is a crucial aspect of human health and environmental sustainability. With increasing pollution and contamination, it is imperative to monitor and predict water quality levels to ensure safe consumption and protect ecosystems. This Water Quality Prediction Web App aims to address this challenge by providing a user-friendly platform for predicting water quality parameters based on various input factors. The app leverages advanced machine learning algorithms to analyse historical data and real-time measurements, enabling accurate predictions of key water quality indicators such as pH, turbidity, dissolved oxygen, and total dissolved solids. By combining data from multiple sources, including sensors, historical records, and environmental factors, the app can provide valuable insights into potential water quality issues and support informed decision-making. The developed web application will utilize a combination of data collection, preprocessing, feature engineering, model selection, training, evaluation, and web development phases. By gathering historical water quality data from reliable sources and preprocessing it to ensure consistency and completeness, the application will lay a solid foundation for accurate predictions. Machine learning algorithms, such as linear regression, decision trees, or neural networks, will be employed to analyse the pre-processed data and establish relationships between water quality parameters and predicted values. The trained models will then be integrated into a user-friendly web interface, allowing users to input water quality data and receive real-time predictions. This web application is designed to be accessible to a wide range of users, from environmental scientists and policymakers to individuals concerned about their local water supply. This web application is designed to be accessible to a wide range of users, from environmental scientists and policymakers to individuals concerned about their local water supply. With its intuitive interface and informative visualizations, the app empowers users to understand water quality trends, identify potential risks, and take proactive measures to protect water resources. Existing water quality monitoring systems may face limitations in terms of data availability, accuracy, and accessibility. They may also struggle to integrate data from diverse sources, such as sensors, historical records, and environmental factors. This can hinder the development of comprehensive and predictive models for water quality assessment.

To address these challenges, a Water Quality Prediction Web App is proposed. This app aims to provide a user-friendly platform for predicting water quality parameters based on various input factors, leveraging advanced machine learning techniques and real-time data integration. By combining data from multiple sources and employing predictive models, the app can offer valuable insights into water quality trends, identify potential risks, and support informed decision-making. With its intuitive interface and informative visualizations, the app empowers users to understand water quality trends, identify potential risks, and take proactive measures to protect water resources.

#### 1.2 PROBLEM DEFINITION

Water quality is a critical issue that affects human health, ecosystems, and economic activities. With increasing population growth, industrialization, and climate change, the pressure on water resources is intensifying, leading to pollution and contamination. By gathering historical water quality data from reliable sources and preprocessing it to ensure consistency and completeness, the application will lay a solid foundation for accurate predictions. Machine learning algorithms, such as linear regression, decision trees, or neural networks, will be employed to analyse the pre-processed data and establish relationships between water quality parameters and predicted values. The trained models will then be integrated into a user-friendly web interface, allowing users to input water quality data and receive real-time predictions. This web application is designed to be accessible to a wide range of users, from environmental scientists and policymakers to individuals concerned about their local water supply. This web application is designed to be accessible to a wide range of users, from environmental scientists and policymakers to individuals concerned about their local water supply. With its intuitive interface and informative visualizations, the app empowers users to understand water quality trends, identify potential risks, and take proactive measures to protect water resources. Existing water quality monitoring systems may face limitations in terms of data availability, accuracy, and accessibility. They may also struggle to integrate data from diverse sources, such as sensors, historical records, and environmental factors. This can hinder the development of comprehensive and predictive models for water quality assessment. Traditional water quality monitoring methods often involve manual sampling and laboratory analysis, which can be time-consuming, expensive, and limited in their coverage The inability to accurately predict water quality can result in severe consequences, such as outbreaks of waterborne diseases, damage to aquatic ecosystems, and economic losses. There is a pressing need for a reliable and efficient system that can monitor water quality in real-time, identify potential risks, and provide timely interventions.

# Chapter 2 LITERATURE SURVEY

## INTRODUCTION

A literature survey or a literature review in a project report is that section which shows various analysis and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of project. Once the programmers start building the tool programmers need a lot of external support. This support can be obtained from senior programmers, books or from the websites. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem of statement. Literature survey is the most important sector in the software development process. Before developing the tools and the associated designing the software it is necessary to determine the survey the time factor, resource requirement etc., The consumer needs regarding online customer service differs from person to person. The needs are also based off each person’s personal needs. We need to identify and anticipate these needs in order to completely and accurately meet them. A comprehensive literature review on water quality prediction web applications reveals significant advancements in leveraging machine learning (ML) models, Internet of Things (IoT) technologies, and cloud computing to monitor and predict water quality in real-time. These technologies aim to enhance water resource management, prevent pollution, and protect public health by offering accurate and timely predictions of water quality parameters. The integration of these tools into web applications provides accessibility and scalability, allowing for more efficient monitoring across various water bodies.

## 2.2 LITERATURE SURVEY

**2.2.1 Water Quality Prediction using Machine Learning**

**Author Name:** Nishant Rawat, Mangani Daudi Kazembe, Pradeep Kumar Mishra

#### Year of Publish: 2022

Freshwater is a critical resource for agriculture and industry's survival. Examination of water quality is a fundamental stage in the administration of freshwater assets. As indicated by the World Health Organization's yearly report, many individuals are getting sick or some are dead due to the lack of safe drinking water, especially pregnant ladies and kids. It is critical to test the quality of water prior to involving it for any reason, whether it is for animal watering, chemical spraying(Pesticides etc), or drinking water. Water quality testing is a strategy for finding clean drinking water. Accordingly, appropriate water monitoring is basic for safe, clean, and sterile water. Water testing is fundamental for looking at the legitimate working of water sources, testing the safety of drinking water, identifying disease outbreaks, and approving methodology and safeguard activities. Water quality is a proportion of a water’s readiness for a specific utilize in view of physical, chemical, and biological qualities.

Water is the principal source for shipping energy to each cell in the body and is additionally the regulator of all body capacities. The cerebrum contains 80% of water. Extreme drying out may prompt mental hindrances and loss of capacity to obviously think. Water is one of the most fundamental regular assets for the endurance of the whole life on this planet. In light of the nature of water, it tends to be utilized for various purposes like drinking, washing, or water system. Plants and creatures likewise rely upon water for their endurance. To put it plainly, all living organic entities need an enormous amount and great nature of water for presence. Freshwater is a fundamental asset to horticulture and industry for its essential presence. Water quality observation is a key stage in the administration of freshwater assets.

**2.2.2 Water quality prediction in the Yellow River source area based on the Deep TCN-GRU model**

**Author Name :** Xijuan Wu, Qiang Zhang, Fei Wen, and Ying Qi

#### Year of Publish : 2022

Water quality prediction is critical in water resource management. Accurate water quality prediction can detect potential water quality issues ahead of time and provides an important scientific foundation for achieving sustainable water resource management. To predict the acid-base index (pH) and total nitrogen content (TN) in water quality indicators, this study uses data from the Kaifeng Yellow River water source area to propose a deep learning combination model (Deep TCN-GRU) that combines the benefits of convolutional neural networks (CNN) and recurrent neural networks (RNN). The study analysed the effects of data processing, different lag values, and different prediction durations on the predictive performance of the model, as well as compared the predictive ability of different deep learning models and explored their predictive performance on water quality data from different water sources. The research results found that data processing can significantly reduce noise in the data and improve the predictive ability of the model; The DeepTCN-GRU model has the best prediction performance for water quality indicators pH and TN when the lag value is 30 days and the prediction duration is 1 day; Compared to other deep learning models, the DeepTCN-GRU model reduces RMSE, MAE, and MSE metrics by at least 29.69 %, 40.21 %, and 36.03 %, R2 There has been a minimum 6.63 % gain in value; In the prediction of water quality data from different water sources using the DeepTCN-GRU model, R2 values are all above 0.9. Overall, the DeepTCN-GRU model provides significant support for Yellow River water quality monitoring and management.

* + 1. **Water quality prediction based on sparse dataset using enhanced machine learning**

**Author Name :** Song, Zhang, and Liu.

#### Year of Publish : 2021

Water quality in surface bodies remains a pressing issue worldwide. While some regions have rich water quality data, less attention is given to areas that lack sufficient data. Therefore, it is crucial to explore novel ways of managing source-oriented [surface water pollution](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/surface-water-pollution) in scenarios with infrequent data collection such as weekly or monthly. Here we showed sparse-dataset-based prediction of [water pollution](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/water-pollution) using machine learning. We investigated the efficacy of a traditional Recurrent Neural Network alongside three Long Short-Term Memory (LSTM) models, integrated with the Load Estimator (LOADEST).The research was conducted at a river-lake confluence, an area with intricate hydrological patterns. The SA-LSTM-LOADEST model improved upon the standalone SA-LSTM model by reducing the [Root Mean Square Error](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/root-mean-square-error) (RMSE) by 24.6% for COD and 21.3% for NH3N.

Surface water is essential for global [biogeochemical cycles](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/biogeochemical-cycle) related to [environmental health](https://www.sciencedirect.com/topics/earth-and-planetary-sciences/environmental-health) and human well-being. Due to the expansion of human activities (e.g., urbanization, agriculture, and dam construction), the impaired water quality of various surface water bodies (e.g., rivers and lakes) has been a major concern worldwide Many lakes and river networks are directly connected, affecting the accumulation and blend of pollutants in surface water bodies. Extreme drying out may prompt mental hindrances and loss of capacity to obviously think. Water is one of the most fundamental regular assets for the endurance of the whole life on this planet. In light of the nature of water, it tends to be utilized for various purposes like drinking, washing, or water system. Plants and creatures likewise rely upon water for their endurance. To put it plainly, all living organic entities need an enormous amount and great nature of water for presence.

* 1. **LITERATURE SURVEY SUMMARY**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Reference** | **Author Name** | **Paper** | **Theme** | **Area of Examination** | **Algorithm** | **Result** |
| 1. | Geetha, Gouthami et al. | Internet of Things Enabled Real Time Water Quality Monitoring System ,2017. | Water quality monitoring | Test water samples and upload data on internet for analysis | None | NA |
| 2. | Ahmed et al. | Efficient Water Quality Prediction Using Supervised Machine Learning, 2019 | Water quality levels | Use of machine learning algorithms to estimate water quality index | Gradient Boost Algorithm | Make a base for an economical ongoing water quality recognition framework. |
| 3. | Ashwini et al. | “Intelligent Model For Predicting Water Quality” | Water quality check | Plan and foster a minimal expense framework for the ongoing observing of water quality utilizing the Internet of Things (IoT) and Machine Learning (ML) | K-Nearest Neighbour | It deliver a practical and economical solution without any human intervention |
| 4. | Prasad et al. | “Smart Water Quality Monitoring System”, 2015 | Water quality monitoring system | Upload water quality data onto the internet using IoT, and wireless sensors | None | Successfully send the alarm based on the parameter for immediate action. |
| 5. | Mohammed et al. | “Machine Learning: Based Detection of Water Contamination in Water Distribution systems”,2018 | Water contamination | Detection of water contamination using machine learning model | None | NA |
| 6. | Singh et al. | Review on Data Mining Techniques for Prediction of Water Quality,2017 | Water quality prediction and data mining | Studying various data mining techniques for prediction of water quality | Naïve Bayes, Back Propagation, KNN | NA |
| 7. | Kumar et al. | Smart Water Monitoring System for Real-Time Water Quality and Usage Monitoring,2018 | Smart Water Quantity meter and Smart Water Quality meter | Configuration Smart Water Quantity Meter to guarantee water protection by observing how much water drank by a family, and informing something very similar to the shopper and the power | None | Implement quality check meter which improve the predict rate and reduce the error. |
| 8. | Koditalaet al. | Water Quality Monitoring System using IoT and  Machine Learning, in Proceedings of the IEEE International Conference on Research in Intelligent and Computing in Engineering, pp.1-5, 2018 | Water quality monitoring | Use of emerging technologies like IoT, machine learning and cloud computing to replace traditional water quality monitoring techniques | None  (but designed some) | Used several sensor to determine the quality of the water which are inexpensive giving a inexpensive solution. |
| 9. | [Haghiabi](javascript:;) et al. | Water quality prediction using machine learning methods, 2018 | Water quality monitoring | Examine execution of artificial intelligence strategies remembering artificial neural network for anticipating water quality parts | Firefly Algorithm | NA |
| 10. | Gollapalli et al. | Ensemble Machine Learning Model to Predict the Waterborne Syndrome, 2022 | Maintain hygienic access to clean water | Use of machine learning model extract data on hygienic conditions and water quality | Naïve Bayes | Address the challenges associated with waterborne disease in low income nation. |

# Chapter 3 SYSTEM ANALYSIS

* 1. **EXISTING SYSTEM**

Existing water quality prediction web apps often rely on traditional statistical methods or rule-based systems. These approaches may have limitations in terms of accuracy, especially when dealing with complex and non-linear relationships between water quality parameters. Additionally, many existing systems lack real-time monitoring capabilities and user-friendly interfaces, hindering their effectiveness and accessibility. While some existing systems incorporate machine learning techniques, they may be limited in terms of the algorithms used and the complexity of the models. Furthermore, the integration of IoT devices and advanced data analytics techniques is not always present in existing systems, limiting their ability to provide comprehensive and up-to-date water quality predictions. Furthermore, many existing systems are not designed to be user-friendly or accessible to a wide range of users. They may require specialized knowledge or technical skills to operate, limiting their usefulness for individuals and organizations outside the field of environmental science. This web application aims to provide a comprehensive solution for monitoring and predicting water quality. By leveraging machine learning algorithms, the system will analyse various water quality parameters to accurately forecast future trends. A user-friendly interface will allow for easy data input and visualization of results. Traditional water quality monitoring often relies on manual sampling and laboratory analysis, which can be time-consuming, expensive, and limited in coverage. While some automated monitoring systems exist, they may face limitations in terms of data availability, accuracy, and accessibility.

## DISADVANTAGES OF EXISTING SYSTEM

* **Data Sparsity and Incomplete Datasets**: Many water quality prediction systems rely on sparse or incomplete datasets, which limit the accuracy and reliability of predictions.
* **Limited Real-Time Capabilities**: Many current systems lack the infrastructure for continuous, real-time monitoring.
* **High Dependency on Costly IoT Infrastructure**: For real-time monitoring, many systems depend on IoT networks, which require substantial investment in sensors and maintenance.
* **Complex Model Interpretability**: Many water quality prediction systems utilize complex machine learning models (e.g., deep learning) that are challenging to interpret.

## PROPOSED SYSTEM

Existing water quality prediction web apps often rely on traditional statistical methods or rule-based systems. These approaches may have limitations in terms of accuracy, especially when dealing with complex and non-linear relationships between water quality parameters. Additionally, many existing systems lack real-time monitoring capabilities and user-friendly interfaces, hindering their effectiveness and accessibility. While some existing systems incorporate machine learning techniques, they may be limited in terms of the algorithms used and the complexity of the models. Furthermore, the integration of IoT devices and advanced data analytics techniques is not always present in existing systems, limiting their ability to provide comprehensive and up-to-date water quality predictions. This web application aims to provide a comprehensive solution for monitoring and predicting water quality. By leveraging machine learning algorithms, the system will analyse various water quality parameters to accurately forecast future trends. A user-friendly interface will allow for easy data input and visualization of results. The application will be equipped with real-time monitoring capabilities, enabling users to stay informed about current water quality conditions. within an image or video, making it suitable for use in detecting and classifying multiple types of accidents, including car crashes, pedestrian accidents, and bicycle accidents.

#### ADVANTAGES OF PROPOSED SYSTEM

* **Real-Time Monitoring and Alerts**: Many water quality prediction applications integrate IoT sensors and cloud systems, enabling real-time data collection and instant analysis.
* **Data-Driven Decision Support**: By leveraging machine learning models, these applications can analyse historical and real-time data to make accurate water quality predictions.
* **Wide Accessibility and Scalability**: Web-based interfaces allow users from various locations to access water quality data from anywhere, fostering greater transparency and public awareness.
* **Cost and Resource Efficiency**: Compared to manual sampling and testing methods, automated water quality monitoring reduces labor costs and resource consumption.

#### FEASIBILITY STUDY

A feasibility study for a water quality prediction web application involves analyzing the project’s technical, economic, and operational viability to ensure successful development and deployment. Technical feasibility assesses the availability and capability of the technology required for real-time data collection, processing, and predictive analysis. This includes IoT sensor networks for data acquisition, machine learning models for prediction, and cloud platforms for data storage and computational needs.

* 1. **HARDWARE ENVIRONMENT**
* Processor : Pentium Dual Core 2.00GH
* Hard disk : 120 GB
* RAM : 2GB (minimum)
* Keyboard : 110 keys enhanced
  1. **SOFTWARE ENVIRONMENT**
* Operating system : Windows7 (with service pack 1), 8, 8.1 ,10 and 11
* Language : Python
  1. **TECHNOLOGIES USED**
* IDE - Visual Studio, Google Colab, Jupyter Notebook
* Framework - Stream-lit
* Machine Learning
  + 1. **Python**

Python is a high-level, interpreted programming language that is widely used in various domains such as web development, data science, artificial intelligence, scientific computing, and more. It was first released in 1991 and has since become one of the most popular programming languages in the world. Some key features of Python include:

* Easy to Learn: Python has a simple and easy-to-learn syntax, which makes it an ideal language for beginners.
* Interpreted Language: Python is an interpreted language, which means that the code is executed line by line, making it easier to test and debug.
* Cross-Platform: Python can be run on various platforms, including Windows, macOS, and Linux.
* Large Standard Library: Python has a large standard library that provides a wide range of built-in modules for various tasks, such as file I/O, regular expressions, networking, and more.
* Open Source: Python is open-source software, which means that the source code is freely available to anyone and can be modified and redistributed.
* Object-Oriented: Python is an object-oriented language, which means that it supports object-oriented programming concepts such as encapsulation, inheritance, and polymorphism.

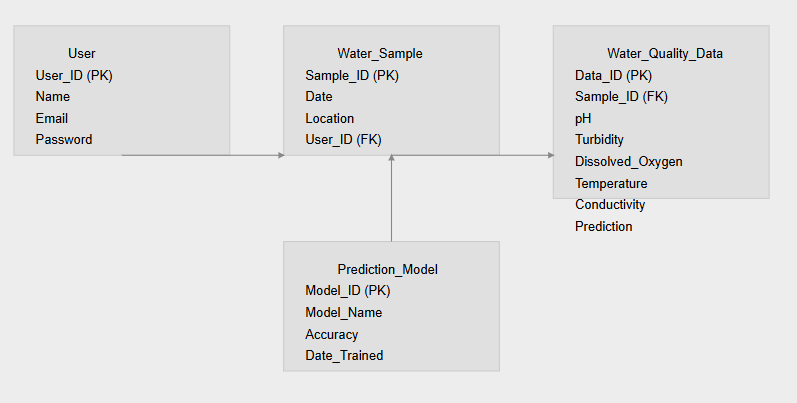
**3.9.2 Machine Learning**

Machine learning algorithms like Random Forest and XG Boost are powerful tools for predicting air quality. Random Forest, an ensemble learning method, creates multiple decision trees and averages their predictions. This approach reduces overfitting and improves accuracy. XG Boost, on the other hand, is a gradient boosting algorithm that iteratively adds decision trees to the model, focusing on correcting errors made by previous trees. This leads to highly accurate and robust predictions. By leveraging historical air quality data, meteorological factors, and other relevant features, these algorithms can effectively model complex patterns and dependencies. This enables the system to accurately forecast air quality levels, identify trends, and issue timely alerts to the public. Additionally, these models can help identify the most significant factors contributing to air pollution, aiding in targeted interventions and policy-making. Random Forest operates by constructing multiple decision trees, each trained on a random subset of the data. XG Boost, on the other hand, is a gradient boosting algorithm that iteratively adds decision trees to the model, focusing on correcting errors made by previous trees. This leads to highly accurate and robust predictions. By leveraging historical air quality data, meteorological factors, and other relevant features, these algorithms can effectively model complex patterns and dependencies. This enables the system to accurately forecast air quality levels, identify trends, and issue timely alerts to the public.

# Chapter 4 SYSTEM DESIGN

#### ENTITY-RELATIONSHIP DIAGRAM

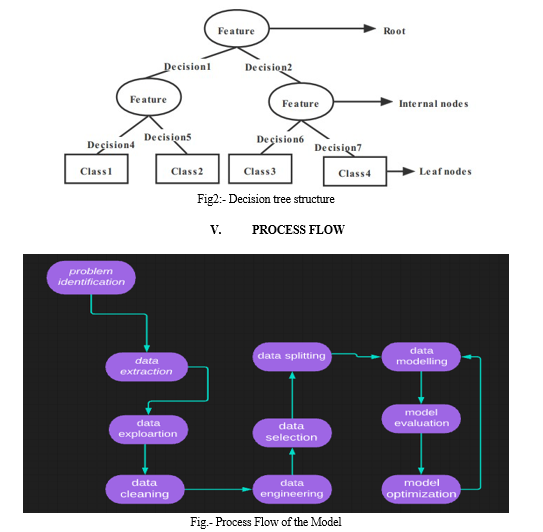
The relationships between database entities can be seen using an entity- relationship diagram (ERD). The entities and relationships depicted in an ERD can have further detail added to them via data object descriptions. In software engineering, conceptual and abstract data descriptions are represented via entity- relationship models (ERMs). Entity-relationship diagrams (ERDs), entity- relationship diagrams (ER), or simply entity diagrams are the terms used to describe the resulting visual representations of data structures that contain relationships between entities. As such, a data flow diagram can serve dual purposes. To demonstrate how data is transformed across the system. To provide an example of the procedures that affect the data flow.

****

**Fig 4.1 Entity Relationship Diagram**

* 1. **DATA FLOW DIAGRAM (DFD)**

The whole system is shown as a single process in a level DFD. Each step in the system's assembly process, including all intermediate steps, are recorded here. The "basic system model" consists of this and 2-level data flow diagrams. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method (SSADM). Superficially, DFDs can resemble flow charts or Unified Modeling Language (UML), but they are not meant to represent details of software logic. DFDs make it easy to depict the business requirements of applications by representing the sequence of process steps and flow of information using a graphical representation or visual representation rather than a textual description.



**Fig 4.1.1Data Flow Diagram**

#### UML DIAGRAMS

* + 1. **Use Case Diagram**

A use case diagram is a type of Unified Modeling Language (UML) diagram that represents the interactions between a system and its actors, and the various use cases that the system supports. It is a visual representation of the functional requirements of the system and the actors that interact with it. Use case diagrams typically include the following elements:

* Actors: Actors are external entities that interact with the system. They can be human users, other systems, or devices.
* Use Cases: Use cases are the specific functions or tasks that the system can perform. Each use case represents a specific interaction between an actor and the system.
* Relationships: Relationships are used to indicate how the actors and use cases are related to each other. The two main relationships in a use case diagram are "uses" and "extends". "Uses" relationship indicates that an actor uses a specific use case, while "extends" relationship indicates that a use case extends or adds functionality to another use case.
* System Boundary: The system boundary is a box that contains all the actors and use cases in the system.



#### Class Diagram

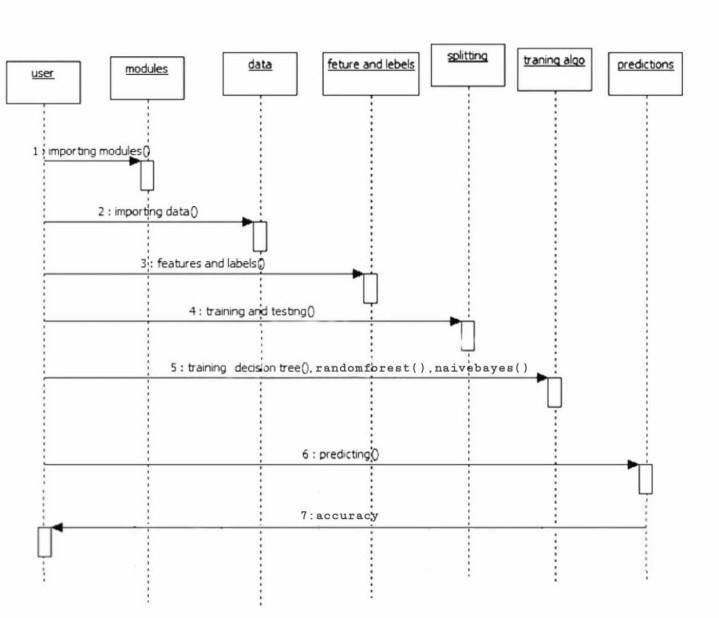
In essence, this is a "context diagram," another name for a contextual diagram. It simply stands for the very highest point, the 0 Level, of the procedure. As a whole, the system is shown as a single process, and the connection to externalities is shown in an abstract manner.

* A + indicates a publicly accessible characteristic or action.
* A - a privately accessible one.
* A # a protected one.
* A - denotes private attributes or operations.

#### 

#### Sequence Diagram

These are another type of interaction-based diagram used to display the workings of the system. They record the conditions under which objects and processes cooperate. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

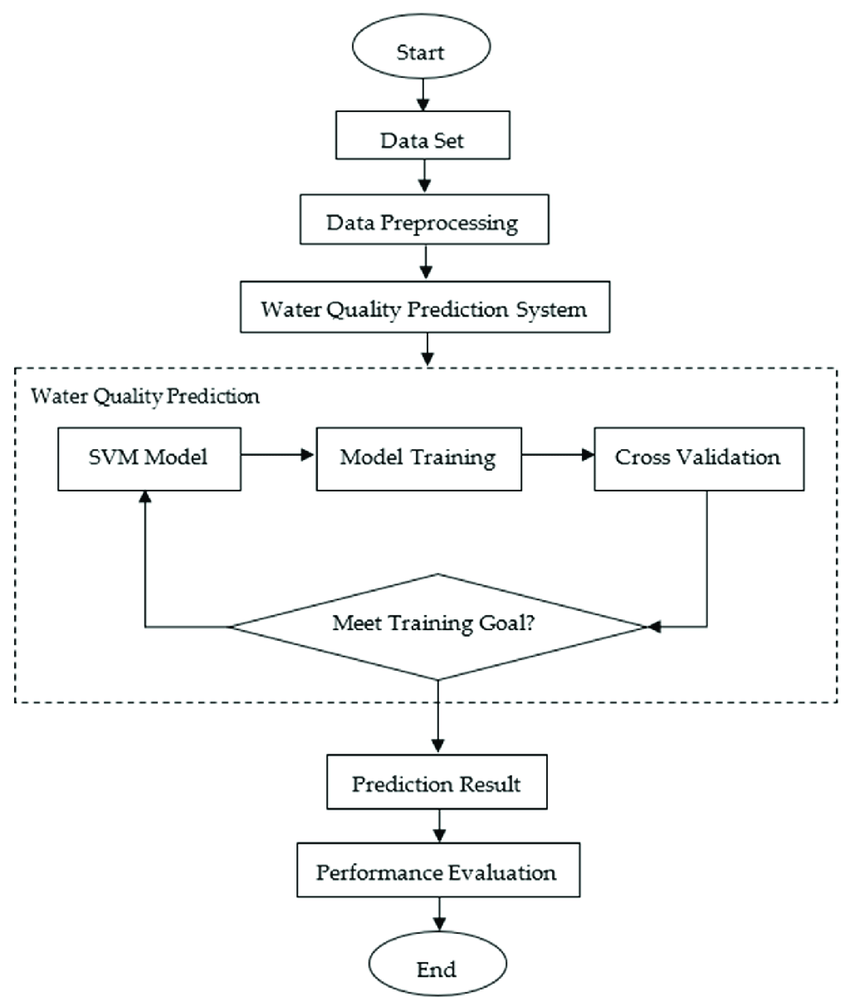


# Chapter 5

**SYSTEM ARCHITECTURE**

## ARCHITECTURE DIAGRAM

This graphic provides a concise and understandable description of all the entities currently integrated into the system. The diagram shows how the many actions and choices are linked together. You might say that the whole process and how it was carried out is a picture. The figure below shows the functional connections between various entities.



XGB Classifier

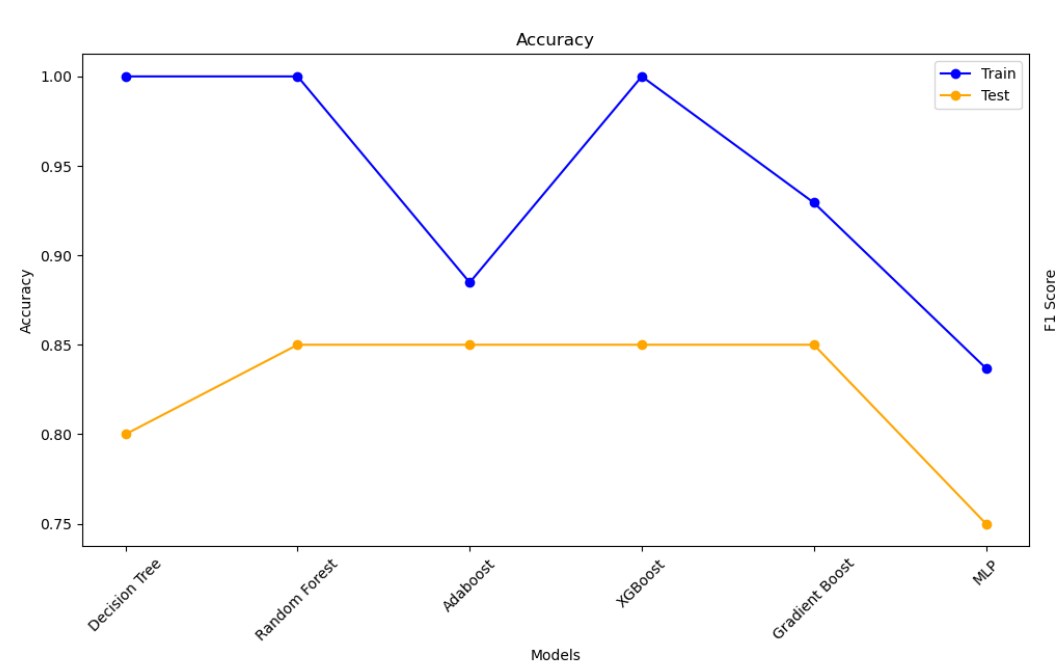
Fig 5.1 Architecture Diagram

This architecture diagram represents a Water Quality Prediction System using the XG Boost (XGB Classifier) model. The workflow begins with data collection, followed by data preprocessing to clean and prepare the dataset for training. The pre- processed data is then passed into the Water Quality Prediction System. In this phase, the XGB Classifier model is initialized and undergoes model training. During training, cross-validation is used to evaluate the model's performance iteratively, ensuring it generalizes well to unseen data. If the model does not meet the desired training criteria, the training and cross-validation steps are repeated with potential hyperparameter tuning. Once the training goal is met, the prediction results are generated. These predictions are then subjected to performance evaluation to assess the model's accuracy, precision, and other metrics. The process concludes after performance evaluation, marking the end of the workflow.

#### ALGORITHMS

* + 1. **XGB Classifier**

XG Boost, which stands for Extreme Gradient Boosting, is a type of ensemble learning method that combines multiple decision trees to make accurate predictions. It combines multiple decision trees to make predictions, improving accuracy and reducing overfitting. XG Boost uses a gradient boosting framework, which means each new tree is trained to correct the errors made by the previous ones. XG Boost incorporates regularization techniques to prevent overfitting and improve generalization. XG Boost is known for its efficiency and can handle large datasets effectively It can handle various types of data, including numerical and categorical features.This adaptation focuses on the iterative training and validation process that is particularly relevant to gradient-boosted decision trees, as implemented in XG Boost, often involving techniques like hyperparameter tuning and early stopping to optimize model performance.



#### Fig 5.2.1 XGB Classifier Water Quality Prediction Model

# Chapter 6

**SYSTEM IMPLEMENTATION**

## MODULE 1: DATA COLLECTION AND PREPROCESSING

In any data-driven application, data quality is paramount, and water quality prediction is no exception. The first step involves collecting historical and real-time water quality data from reliable sources such as government agencies (e.g., the Environmental Protection Agency), research institutions, or environmental monitoring stations. This data might include parameters like pH level, turbidity, dissolved oxygen, temperature, and other chemical or biological indicators that reflect the state of water quality. Once collected, data preprocessing is essential. Real-world data often contains missing values, outliers, and inconsistencies due to sensor malfunctions, human errors, or environmental variability. Handling these issues involves techniques like filling missing values with median or mean values, removing or capping outliers, and ensuring uniform data formats. Outliers that indicate unusual events are carefully examined; they may either be corrected if erroneous or retained if they signify significant environmental changes. Feature selection and transformation are crucial for improving model performance. Not all data points might contribute to predictive power, so only the most relevant features are chosen, possibly informed by domain knowledge. Transformation processes like scaling or normalizing numeric values help the model learn patterns more efficiently. In some cases, derived parameters might be calculated (e.g., Biochemical Oxygen Demand (BOD) based on existing metrics), adding valuable insights into water quality trends. This thorough preprocessing stage helps ensure that the model has clean, reliable data to learn from, which is crucial for accurate predictions.In the preprocessing phase, data imputation strategies are carefully chosen based on the nature of missing values. For instance, missing pH values may be imputed using median values for similar locations, while missing turbidity readings might be estimated using interpolation techniques if they form part of a continuous time series. In cases where missing values are too frequent for a specific parameter, that parameter may be excluded from the model to avoid introducing noise. Outlier detection is equally important, as certain data points could be anomalies due to temporary environmental events or sensor errors. Outliers are detected through statistical techniques like Z-scores or the interquartile range (IQR) method, and appropriate action is taken—either removing them or adjusting based on their significance.

## MODULE 2: MODEL TRAINING

The next step in building the water quality prediction application is model development and training. Based on the characteristics of the data and desired prediction accuracy, an appropriate machine learning algorithm is selected. Factors such as the dataset’s size, the complexity of relationships between parameters, and whether the task is classification (e.g., labelling water quality as "Good," "Moderate," or "Poor") or regression (predicting specific parameter values) influence this choice. Algorithms commonly used include Random Forest, Support Vector Machines, or neural networks, each with distinct advantages depending on the prediction goals. Once the algorithm is chosen, the model is trained using the pre-processed data. Training involves feeding the model historical water quality data along with known outcomes, enabling it to learn patterns and establish relationships between input features and target outputs. This phase often requires splitting the dataset into training, validation, and test sets. The training set teaches the model, while the validation set helps tune hyperparameters, ensuring optimal performance. The test set provides an unbiased evaluation of the model’s accuracy and generalization capability. Throughout this process, techniques like cross-validation and parameter tuning are employed to avoid overfitting and optimize the model’s performance. Overfitting occurs when the model memorizes the training data rather than generalizing patterns, which can lead to inaccurate predictions on new data. The trained model is then evaluated for metrics such as accuracy, precision, recall, and F1 score, which quantify how well it can predict water quality. During the model training phase, the application of techniques like grid search or random search can help optimize hyperparameters, ensuring the model performs at its best. Hyperparameters like the depth of trees, the number of estimators, and the minimum samples split for the Random Forest model are tuned to balance bias and variance. Cross-validation further ensures that the model is tested on multiple subsets of data, helping confirm its robustness and preventing overfitting. For further refinement, ensemble learning methods may be explored, combining predictions from Random Forest with other models like Gradient Boosting or XG Boost to improve overall accuracy. These advanced techniques help in situations where single models have limitations, enhancing predictive power and resilience to variations in input data. In addition, the model's output may be validated against established water quality guidelines, like those from the World Health Organization (WHO) or local environmental standards. This validation helps interpret the model's predictions within a real-world context, ensuring that the outputs align with actionable water quality levels and thresholds.

## MODULE 3: PREDICTION OF OUTPUT

The **XG Boost (Extreme Gradient Boosting)** classifier is a powerful and flexible machine learning algorithm widely used in classification and regression tasks. It builds on the principles of gradient boosting, a technique that sequentially builds decision trees to correct the errors of previous ones, thereby improving accuracy. XG Boost stands out for its optimized implementation, offering high computational efficiency, fast processing times, and exceptional accuracy, which makes it suitable for complex datasets, like water quality data, where there may be nonlinear relationships and intricate patterns among parameters. In the water quality prediction system, the XG Boost classifier can be particularly effective for handling imbalanced and noisy datasets. Since it is robust to outliers and capable of managing both continuous and categorical variables, XG Boost is ideal for datasets with varied water quality indicators (e.g., pH, turbidity, dissolved oxygen, temperature, etc.). This adaptability makes it well-suited to handle different environmental factors that influence water quality, such as seasonal shifts, pollution events, or changing weather conditions. XG Boost uses a regularized objective function, which includes both a loss function (for error minimization) and a regularization term that helps prevent overfitting. Regularization makes the model simpler and improves its generalization capability on new data, which is crucial for accurate water quality predictions. This regularization is controlled by hyperparameters like lambda and alpha, which are tuned during model training to optimize performance and minimize errors. After training and evaluating the XG Boost model, the next step is to integrate it into the web application. The model is saved using serialization techniques (e.g. pickle) to maintain compatibility with the web application’s backend. The serialized model can then be loaded whenever a prediction is requested, allowing for real-time prediction on new user-input water quality data. A RESTful API is developed to handle data input and model inference. The API allows users to submit parameters (e.g., pH, turbidity, dissolved oxygen) and receive a prediction based on the trained XG Boost model. The web application is designed not only to display predictions but also to educate users on water quality indicators. A user-friendly interface is created with clear navigation, allowing users to easily understand the implications of the model’s predictions. Visual elements, such as color-coded indicators or water quality health messages (e.g., "Safe," "Moderate," "Unsafe"), provide immediate insights. Backend integration is also optimized for handling multiple user requests, ensuring scalability. Technologies like Flask, Django, or Fast API facilitate this by providing efficient routing and API management.

# Chapter 7 SYSTEM TESTING

#### BLACK BOX TESTING

#### The user is not given access to or knowledge of the internal workings or details of the data item under test during this type of testing. This approach does not require prior knowledge of the design or the code; instead, test cases are created or constructed only based on the input and output values. The testers are just aware of what is believed to be possible; they are unaware of how it accomplishes this. For instance, we test the web pages using a browser, authorize the input, and then test and validate the outputs against the desired outcome without knowing anything about the inner workings of the website.



Fig 7.1 Black Box Testing

For example, without having any knowledge of the inner workings of the website, we test the web pages by using a browser, then we authorize the input, and last, we test and validate the outputs against the intended result.

#### WHITE BOX TESTING

During this kind of testing, the user is aware of the internal structure and details of the data item, or they have access to such information. In this process, test cases are constructed by referring to the code. Programming is extremely knowledgeable of the manner in which the application of knowledge is significant. White Box Testing is so called because, as we all know, in the tester's eyes it appears to be a white box, and on the inside, everyone can see clearly.This is how the testing got its name.

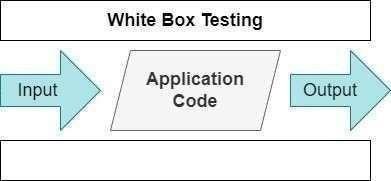


Fig 7.2 White Box Testing

As an instance, a tester and a developer examine the code that is implemented in each field of a website, determine which inputs are acceptable and which are not, and then check the output to ensure it produces the desired result. In addition, the decision is reached by analyzing the code that is really used.

* 1. **TEST CASES**

**TEST REPORT: 01**

**PRODUCT:**  WATE QUALITY PREDICTION BY A MODEL AND DEPLOYING IT AS A WEB APPLICATION

**USE CASE:** UPLOAD DATA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **Action** | **Input** | **Expected output** | **Actual output** | **Test Result** |
| 1 | Verify data retrieval from water quality sample | Location: River A | Display current water quality metrics for River A | [Actual metrics] for River A | Pass/Fail |
| 2 | Test prediction model accuracy | Historical water quality data | Prediction within 10% of actual water quality values | Prediction accuracy [actual %] | Pass/Fail |
| 3 | Check alert generation for unsafe water conditions | pH: 3.5, Turbidity: High | System sends alert for unsafe water conditions | Alert message sent | Pass/Fail |
| 4 | Ensure no alert for safe water conditions | pH: 7.2, Turbidity: Low | No alert is sent | No alert generated | Pass/Fail |
| 5 | Verify model deployment on web application | Access URL | Web app loads successfully with prediction model | Web app loaded [actual status] | Pass/Fail |

**Table-7.3.1 Test Case Dataset Upload**

## TEST REPORT: 02

**PRODUCT:**  WATE QUALITY PREDICTION BY A MODEL AND DEPLOYING IT AS A WEB APPLICATION

**USE CASE:** PREDICTING DATA

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.NO | Action | Input | Expected output | Actual output | Test Result |
| 1 | Test prediction request through web app | User inputs water quality metrics | Model provides water quality prediction | Prediction displayed [actual output] | Pass/Fail |
| 2 | Check response time of prediction model on web app | User submits input via web form | Prediction response time within 3 seconds | Response time [actual time] | Pass/Fail |
| 3 | Verify user interface for displaying predictions | Open web app | Display predictions in readable format on dashboard | Predictions displayed correctly | Pass/Fail |
| 4 | Test alert customization for different water types | User preferences: Custom thresholds | Alert generated based on user-defined thresholds | Alert generated as per user threshold | Pass/Fail |
| 50 | Ensure real-time data integration on web app | Real-time water quality data feed | Data updated every 5 minutes | Data updated every [actual interval] | Pass/Fail |

**Table-7.3.2 Test Case for Predicting Result**

# Chapter 8

**CONCLUSION AND FUTURE ENHANCEMENT**

## CONCLUSION

The water quality prediction web application project has provided a comprehensive solution to a critical environmental and public health issue: monitoring and assessing water quality. This project, which integrates machine learning algorithms like XG Boost with web technologies, has demonstrated how predictive analytics can be harnessed to offer timely, data-driven insights into water safety. With increasing concerns about pollution and environmental degradation, such a tool is essential for helping communities, policymakers, and environmental organizations to take proactive steps in safeguarding water resources. The data-driven nature of the project allows it to leverage historical and real-time data, creating a solid foundation for accurate predictions. By systematically collecting, cleaning, and preprocessing data from reliable sources (such as government agencies and research institutions), the project ensures that the inputs to the model are high-quality and reliable. Preprocessing steps like handling missing values, removing outliers, and scaling features have been instrumental in enhancing model performance. These steps lay the groundwork for a robust prediction system that can respond dynamically to new data and adapt to changes in water quality trends. At the core of this application is the XG Boost machine learning model, a powerful, efficient, and scalable algorithm that is particularly well-suited to handling complex, imbalanced datasets like those in water quality analysis. The project has shown how XG Boost’s capabilities for handling missing data, conducting feature importance analysis, and optimizing performance through regularization can be leveraged to improve predictive accuracy. Through systematic hyperparameter tuning, cross-validation, and the use of metrics such as accuracy, precision, recall, F1 score, and AUC-ROC, the model has been fine-tuned to achieve high performance. This makes it capable of accurately classifying water quality conditions, distinguishing between "Safe," "Moderate," and "Unsafe" levels, and providing valuable insights to users. The web application interface adds a layer of accessibility and usability, bringing the power of machine learning to end users without requiring technical expertise. By creating a simple and intuitive interface, the project allows users—such as environmental scientists, public health officials, and even concerned citizens—to input data, view predictions, and make informed decisions about water quality.

## FUTURE ENCHANCEMENT

The water quality prediction web application holds vast potential for future enhancements to improve accuracy, usability, and scalability. A significant upgrade would be the integration of IoT-based real-time sensor data from various water sources, which could greatly improve the timeliness and precision of predictions. Real-time monitoring of parameters like pH, turbidity, dissolved oxygen, and temperature would allow for immediate data input and more responsive alert systems, enabling communities and agencies to act swiftly if water quality deteriorates unexpectedly. Additionally, the application could benefit from exploring advanced machine learning models beyond XG Boost. For example, deep learning models such as Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks are well-suited for time-series data and could enhance the model’s ability to detect temporal patterns in water quality variations. Hybrid models combining multiple algorithms could also be tested to improve overall performance and handle complex datasets with high dimensionality. Incorporating geospatial analysis would further increase the application’s utility, allowing it to provide location-based insights into water quality trends. By mapping water quality data across different regions, users could visualize geographic patterns of pollution, detect hotspots, and gain a better understanding of environmental factors affecting specific areas. This would be especially beneficial for larger organizations or government agencies that manage water resources across wide areas. The addition of predictive analytics for specific contaminants, such as heavy metals or agricultural runoff, would make the application more granular and informative, enabling targeted interventions and more precise monitoring. Moreover, expanding the application’s accessibility by creating a mobile version could significantly broaden its reach. A mobile app with push notifications for alerting users about sudden changes in water quality would make the tool more convenient and enable real-time alerts for users in remote areas or field locations. Finally, integrating a periodic model retraining mechanism would allow the application to adapt continuously to new data, ensuring its accuracy remains high over time. As more data is collected, the model could be retrained to reflect emerging trends or shifts in environmental conditions, which would increase its reliability and relevance in the long term. These enhancements would position the water quality prediction application as a powerful, adaptable tool capable of serving diverse user groups, from local communities to large organizations, while contributing to improved water quality management and public health outcomes.

# Chapter 9

**APPENDIX 1 – SAMPLE CODING**

## WQPWA.ipynb:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

import plotly.express as px

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import AdaBoostClassifier

from sklearn.ensemble import GradientBoostingClassifier

from XG Boost import XGBClassifier

from sklearn.neural\_network import MLPClassifier

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import train\_test\_split

from sklearn.model\_selection import RandomizedSearchCV, GridSearchCV

from sklearn.metrics import classification\_report

from sklearn.metrics import accuracy\_score

from sklearn.metrics import precision\_score, recall\_score, confusion\_matrix,f1\_score

from warnings import filterwarnings

df = pd.read\_csv('water\_potability.csv’)

df

df.isnull().sum()

df.dropna(how='all', inplace=True)

df.info()

df.Potability.value\_counts()

idx1 = df.query('Potability == 1')['ph'][df.ph.isna()].index

df.loc[idx1, 'ph'] = df.query('Potability == 1')['ph'][df.ph.notna()].mean()

idx0 = df.query('Potability == 0')['ph'][df.ph.isna()].index

df.loc[idx0,'ph'] = df.query('Potability==0')['ph'][df.ph.notna()].mean()

idx1 = df.query('Potability == 1')['Sulfate'][df.Sulfate.isna()].index

df.loc[idx1, 'Sulfate'] = df.query('Potability == 1')['Sulfate'][df.Sulfate.notna()].mean()

idx0 = df.query('Potability == 0')['Sulfate'][df.Sulfate.isna()].index

df.loc[idx0,'Sulfate'] = df.query('Potability==0')['Sulfate'][df.Sulfate.notna()].mean()

idx1 = df.query('Potability == 1')['Trihalomethanes'][df.Trihalomethanes.isna()].index

df.loc[idx1, 'Trihalomethanes'] = df.query('Potability == 1')['Trihalomethanes'][df.Trihalomethanes.notna()].mean()

idx0 = df.query('Potability == 0')['Trihalomethanes'][df.Trihalomethanes.isna()].index

df.loc[idx0,'Trihalomethanes'] = df.query('Potability==0')['Trihalomethanes'][df.Trihalomethanes.notna()].mean()

df.loc[~df.ph.between(6.5, 8.5), 'Potability'] = 0

d = pd.DataFrame(df["Potability"].value\_counts())

fig = px.pie(d, values = "count", names = ["Not Potable", "Potable"], hole = 0.4, opacity = 0.8,

labels = {"label":"Potability","Potability":"Number of Samples"})

fig.update\_layout(title = dict(text = "Pie Chart of Potability Feature"))

fig.update\_traces(textposition = "outside", textinfo = "percent+label")

fig.show()

# Correlation Between Features

df.corr()

sns.clustermap(df.corr(), cmap = "vlag", annot = True, figsize = (6,6))

plt.show()

# Distrubition of Features

filterwarnings('ignore')

non\_potable = df.query("Potability == 0")

potable = df.query("Potability == 1")

plt.figure(figsize = (15,15))

for ax,col in enumerate(df.columns[:9]):

plt.subplot(3,3, ax + 1)

plt.title(col)

sns.kdeplot(x = non\_potable[col], label = "Non Potable")

sns.kdeplot(x = potable[col], label = "Potable")

plt.legend()

plt.tight\_layout()

variables = df.columns

fig, axes = plt.subplots(nrows=3, ncols=3, figsize=(15, 15))

axes = axes.flatten()

for i, var in enumerate(variables):

if(var != 'Potability'):

ax = axes[i]

sns.histplot(df[var], kde=True, ax=ax)

ax.axvline(df[var].mean(), color='red', linestyle='--', label='Mean')

ax.axvline(df[var].median(), color='blue', linestyle='--', label='Median')

ax.annotate(f'Mean: {df[var].mean():.2f}\nMedian: {df[var].median():.2f}',

xy=(0.05, 0.95), xycoords='axes fraction', ha='left', va='top')

ax.set\_title(var)

ax.set\_xlabel(var)

ax.legend()

plt.tight\_layout()

plt.show()

columns = df.columns

fig, ax = plt.subplots(9, 1, figsize=(10, 20))

fig.subplots\_adjust(hspace=0.75)

for i,col in enumerate(columns):

if col != 'Potability':

sns.boxplot(x=col, data=df, ax=ax[i] , color = 'cyan')

plt.show()

def gini(actual, pred):

assert (len(actual) == len(pred))

all = np.asarray(np.c\_[actual, pred, np.arange(len(actual))], dtype=np.float64)

all = all[np.lexsort((all[:, 2], -1 \* all[:, 1]))]

totalLosses = all[:, 0].sum()

giniSum = all[:, 0].cumsum().sum() / totalLosses

giniSum -= (len(actual) + 1) / 2.

return giniSum / len(actual)

x=df[['ph','Hardness','Solids','Chloramines','Sulfate','Conductivity','Organic\_carbon','Trihalomethanes','Turbidity']]

y=df['Potability']

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x,y,test\_size=20,random\_state = 42)

sta = StandardScaler()

x\_train = sta.fit\_transform(x\_train)

x\_test = sta.transform(x\_test)

models={

"Decision Tree": DecisionTreeClassifier(),

"Random Forest": RandomForestClassifier(),

"Adaboost": AdaBoostClassifier(),

"XG Boost": XGBClassifier(),

"Gradient boost":GradientBoostingClassifier(),

"MLP ":MLPClassifier() }

for i in range(len(list(models))):

model=list(models.values())[i]

model.fit(x\_train,y\_train)

y\_train\_pred=model.predict(x\_train)

y\_test\_pred=model.predict(x\_test)

#train

model\_train\_accuracy = accuracy\_score(y\_train,y\_train\_pred)

model\_train\_f1=f1\_score(y\_train,y\_train\_pred,average='weighted')

model\_train\_precision=precision\_score(y\_train,y\_train\_pred)

model\_train\_recall=recall\_score(y\_train,y\_train\_pred)

model\_train\_gini=gini(y\_train,y\_train\_pred)

#test

model\_test\_accuracy = accuracy\_score(y\_test,y\_test\_pred)

model\_test\_f1=f1\_score(y\_test,y\_test\_pred,average='weighted')

model\_test\_precision=precision\_score(y\_test,y\_test\_pred)

model\_test\_recall=recall\_score(y\_test,y\_test\_pred)

model\_test\_gini=gini(y\_test,y\_test\_pred)

print(list(models.keys())[i])

print('-----------------------------------------')

print("Model Performance for Training DataSet")

print("accuracy : ",(model\_train\_accuracy))

print('f1 score : ',(model\_train\_f1))

print('precision : ',(model\_train\_precision))

print('recall : ',(model\_train\_recall))

print('Gini Index : ',(model\_train\_gini))

print('-----------------------------------------')

print("Model Performance for Test DataSet")

print("accuracy : ",(model\_test\_accuracy))

print('f1 score : ',(model\_test\_f1))

print('precision : ',(model\_test\_precision))

print('recall : ',(model\_test\_recall))

print('Gini Index : ',(model\_test\_gini))

print('-----------------------------------------')

print('\n')

import matplotlib.pyplot as plt

# Model names and metrics based on provided data

model\_names = ["Decision Tree", "Random Forest", "Adaboost", "XG Boost", "Gradient Boost", "MLP"]

train\_accuracies = [1.0, 1.0, 0.8848, 1.0, 0.9294, 0.8369]

test\_accuracies = [0.8, 0.85, 0.85, 0.85, 0.85, 0.75]

train\_f1\_scores = [1.0, 1.0, 0.8817, 1.0, 0.9273, 0.8277]

test\_f1\_scores = [0.8, 0.8114, 0.8114, 0.8416, 0.8114, 0.7359]

train\_precisions = [1.0, 1.0, 0.7926, 1.0, 0.9084, 0.7020]

test\_precisions = [0.5, 1.0, 1.0, 0.6667, 1.0, 0.3333]

train\_recalls = [1.0, 1.0, 0.6825, 1.0, 0.7738, 0.5172]

test\_recalls = [0.5, 0.25, 0.25, 0.5, 0.25, 0.25]

train\_ginis = [0.3839, 0.3839, 0.2415, 0.3839, 0.2893, 0.1799]

test\_ginis = [0.15, 0.075, 0.075, 0.2375, 0.075, 0.15]

# Plotting each metric

metrics = {

'Accuracy': (train\_accuracies, test\_accuracies),

'F1 Score': (train\_f1\_scores, test\_f1\_scores),

'Precision': (train\_precisions, test\_precisions),

'Recall': (train\_recalls, test\_recalls),

'Gini Index': (train\_ginis, test\_ginis)

}

# Create a 3x2 grid of subplots

fig, axes = plt.subplots(nrows=3, ncols=2, figsize=(20, 20))

fig.suptitle("Model Performance Comparison", fontsize=16)

# Plot each metric

for ax, (metric, (train\_metric, test\_metric)) in zip(axes.flat, metrics.items()):

ax.plot(model\_names, train\_metric, label="Train", marker='o', color='blue')

ax.plot(model\_names, test\_metric, label="Test", marker='o', color='orange')

ax.set\_title(metric)

ax.set\_xlabel("Models")

ax.set\_ylabel(metric)

ax.set\_xticklabels(model\_names, rotation=45)

ax.legend()

plt.tight\_layout(rect=[0, 0, 1, 0.96]) # Adjust layout to fit title

plt.show()

import pickle

with open('xgb\_model.pkl', 'wb') as f:

pickle.dump(model, f)

**App.py**

import pickle

import numpy as np

from flask import Flask, request, render\_template

# Initialize Flask app

app = Flask(\_\_name\_\_)

# Load the trained XGBClassifier model

with open('water\_quality\_model.pkl', 'rb') as model\_file:

model = pickle.load(model\_file)

@app.route('/')

def home():

return render\_template('index.html')

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

def predict():

if request.method == 'POST':

# Extract input values from the form

ph = float(request.form['ph'])

hardness = float(request.form['Hardness'])

solids = float(request.form['Solids'])

chloramines = float(request.form['Chloramines'])

sulfate = float(request.form['Sulfate'])

conductivity = float(request.form['Conductivity'])

organic\_carbon = float(request.form['Organic\_carbon'])

trihalomethanes = float(request.form['Trihalomethanes'])

turbidity = float(request.form['Turbidity'])

# Prepare the input array for prediction

input\_data = np.array([[ph, hardness, solids, chloramines, sulfate,

conductivity, organic\_carbon, trihalomethanes, turbidity]])

# Make prediction using the loaded model

prediction = model.predict(input\_data)

# Map the prediction (0 or 1) to a label for clarity

result = "Potable" if prediction[0] == 1 else "Not Potable"

# Return the prediction result

return render\_template('result.html', prediction=result)

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=True)

**index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Water Quality Prediction</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

    <style>

        \* {

            margin: 0;

            padding: 0;

            box-sizing: border-box;

        }

        body {

            font-family: 'Arial', sans-serif;

            background-color: #f0f4f8;

            color: #333;

        }

        .container {

            width: 60%;

            margin: 50px auto;

            padding: 30px;

            background-color: white;

            border-radius: 10px;

            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

        }

        h2 {

            text-align: center;

            color: #4CAF50;

            font-size: 2rem;

            margin-bottom: 20px;

        }

        label {

            font-size: 1.1rem;

            color: #555;

        }

        input[type="number"], input[type="text"] {

            width: 100%;

            padding: 12px;

            margin: 10px 0;

            border: 1px solid #ccc;

            border-radius: 5px;

            font-size: 1rem;

        }

        button {

            width: 100%;

            padding: 15px;

            background-color: #4CAF50;

            color: white;

            font-size: 1.2rem;

            border: none;

            cursor: pointer;

            border-radius: 5px;

        }

        button:hover {

            background-color: #45a049;

        }

        .form-group {

            margin-bottom: 20px;

        }

        .footer {

            text-align: center;

            font-size: 0.9rem;

            color: #777;

            margin-top: 30px;

        }

        .footer a {

            text-decoration: none;

            color: #4CAF50;

        }

        .footer a:hover {

            color: #45a049;

        }

    </style>

</head>

<body>

<div class="container">

    <h2>Water Quality Prediction</h2>

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

        <div class="form-group">

            <label for="ph">pH</label>

            <input type="number" step="any" name="ph" required>

        </div>

        <div class="form-group">

            <label for="Hardness">Hardness</label>

            <input type="number" step="any" name="Hardness" required>

        </div>

        <div class="form-group">

            <label for="Solids">Solids</label>

            <input type="number" step="any" name="Solids" required>

        </div>

        <div class="form-group">

            <label for="Chloramines">Chloramines</label>

            <input type="number" step="any" name="Chloramines" required>

        </div>

        <div class="form-group">

            <label for="Sulfate">Sulfate</label>

            <input type="number" step="any" name="Sulfate" required>

        </div>

        <div class="form-group">

            <label for="Conductivity">Conductivity</label>

            <input type="number" step="any" name="Conductivity" required>

        </div>

        <div class="form-group">

            <label for="Organic\_carbon">Organic Carbon</label>

            <input type="number" step="any" name="Organic\_carbon" required>

        </div>

        <div class="form-group">

            <label for="Trihalomethanes">Trihalomethanes</label>

            <input type="number" step="any" name="Trihalomethanes" required>

        </div>

        <div class="form-group">

            <label for="Turbidity">Turbidity</label>

            <input type="number" step="any" name="Turbidity" required>

        </div>

        <button type="submit">Predict Potability</button>

    </form>

</div>

<div class="footer">

    <p>Powered by <a href="#">Water Quality AI</a></p>

</div>

</body>

</html>

**result.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Prediction Result</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.4/css/all.min.css">

    <style>

        \* {

            margin: 0;

            padding: 0;

            box-sizing: border-box;

        }

        body {

            font-family: 'Arial', sans-serif;

            background-color: #f0f4f8;

            color: #333;

        }

        .container {

            width: 60%;

            margin: 50px auto;

            padding: 30px;

            background-color: white;

            border-radius: 10px;

            box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);

        }

        h2 {

            text-align: center;

            color: #4CAF50;

            font-size: 2rem;

            margin-bottom: 20px;

        }

        .result {

            font-size: 1.5rem;

            text-align: center;

            margin: 20px 0;

            padding: 20px;

            background-color: #f8f8f8;

            border-radius: 8px;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        .result p {

            font-size: 1.3rem;

            font-weight: bold;

            color: #4CAF50;

        }

        .result p.not-potable {

            color: #e74c3c;

        }

        .footer {

            text-align: center;

            font-size: 0.9rem;

            color: #777;

            margin-top: 30px;

        }

        .footer a {

            text-decoration: none;

            color: #4CAF50;

        }

        .footer a:hover {

            color: #45a049;

        }

    </style>

</head>

<body>

<div class="container">

    <h2>Water Quality Prediction Result</h2>

    <div class="result">

        <p class="{{ 'not-potable' if prediction == 'Not Potable' else '' }}">

            The water is: <strong>{{ prediction }}</strong>

        </p>

    </div>

    <a href="/" class="button">Go Back</a>

</div>

<div class="footer">

    <p>Powered by <a href="#">Water Quality AI</a></p>

</div>

</body>

</html>

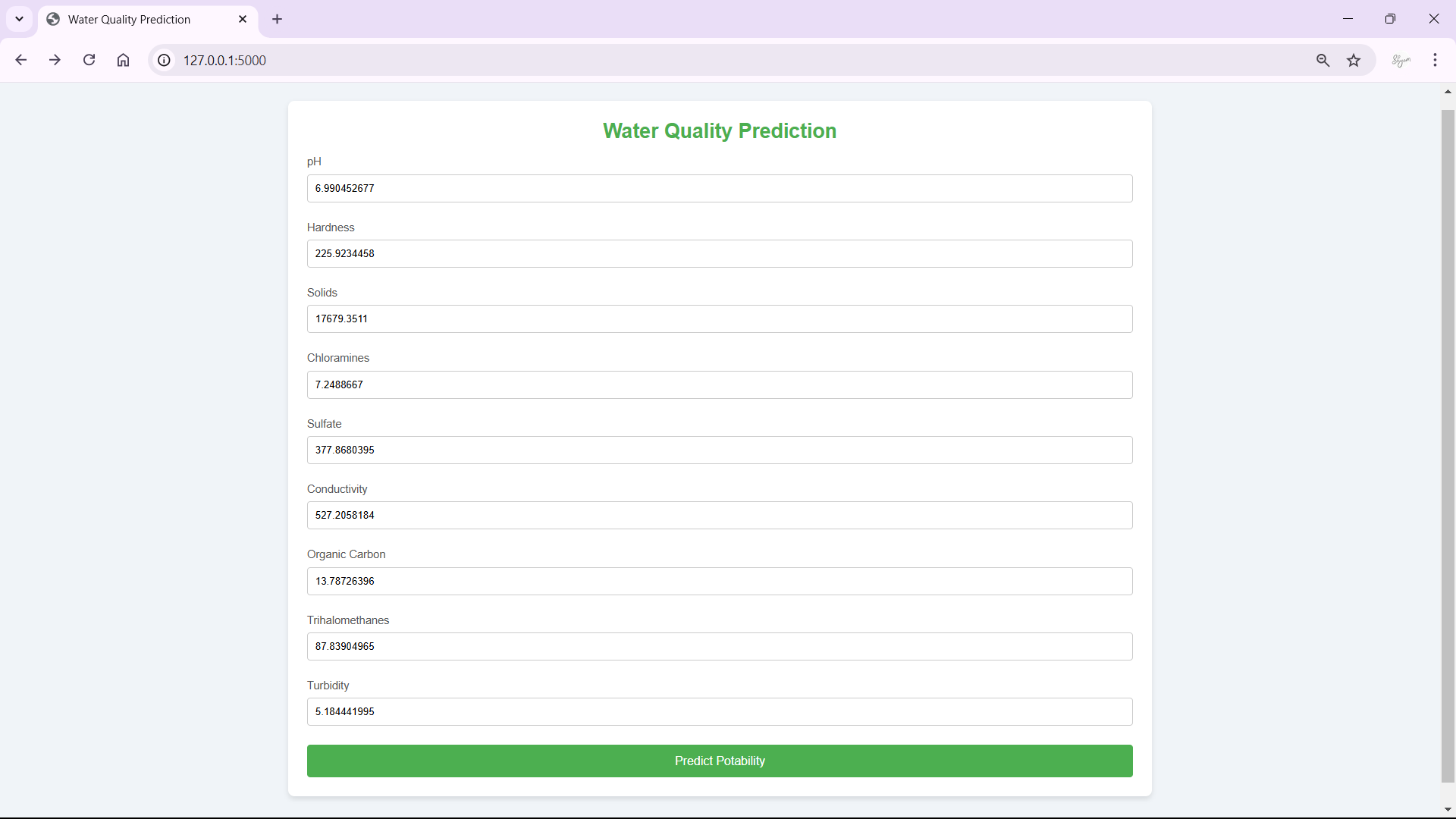
# Chapter 10

# APPENDIX 2 – SAMPLE OUTPUT

## 10.1 Home Page

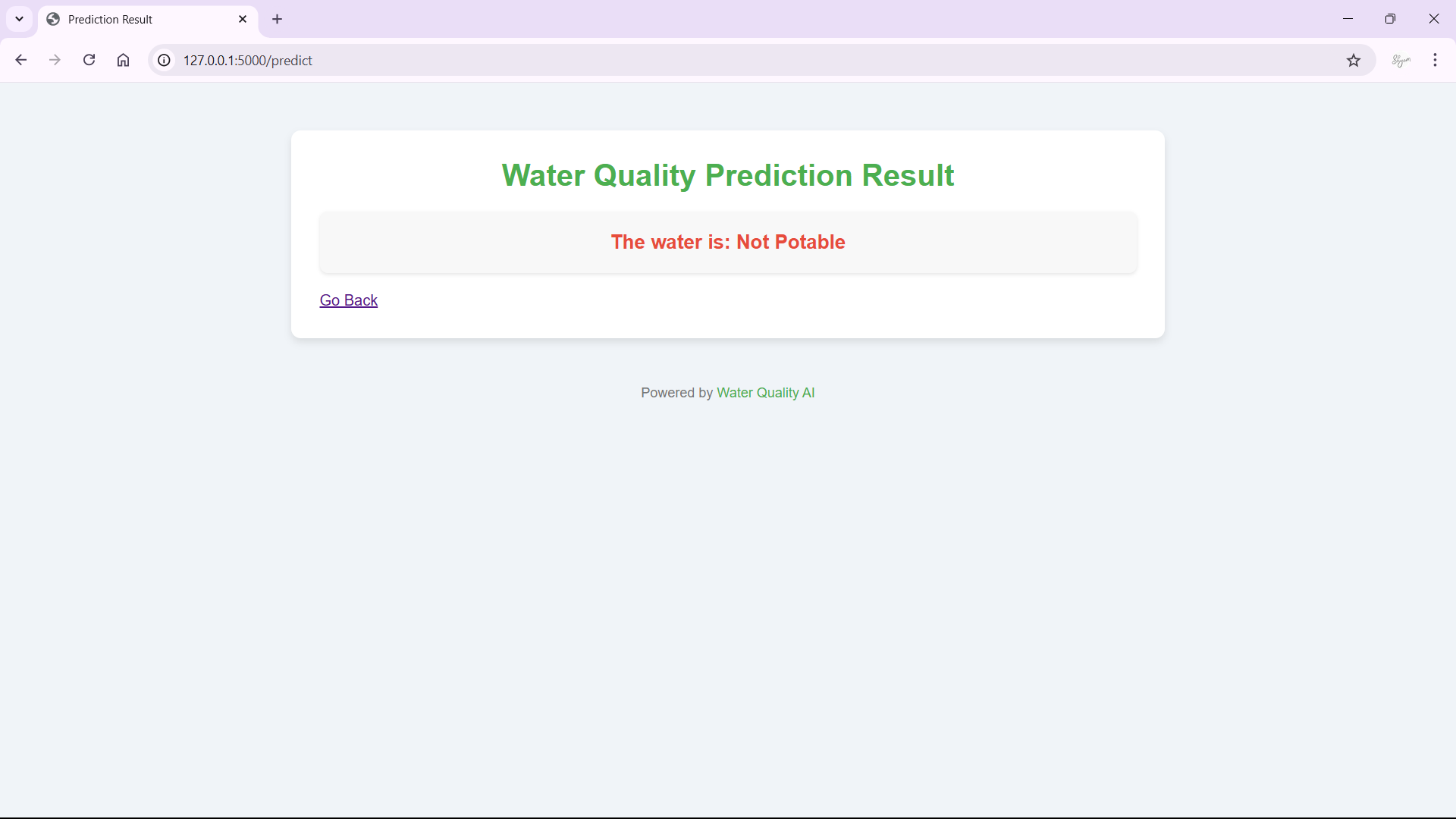
The project output screenshots are shown as follows:

#### Homepage



## The image displays a user interface for a "Water Quality Prediction" tool, designed to help users assess the potability of water based on various chemical and physical parameters. The interface includes labeled input fields for different water quality metrics, such as pH, which indicates acidity or alkalinity; Hardness, measuring calcium and magnesium concentration; Solids, representing the total dissolved solids (TDS); Chloramines, showing the amount of chloramine used as a disinfectant; Sulfate, indicating sulfate ion concentration; Conductivity, reflecting the water's ability to conduct electricity due to ions; Organic Carbon, which measures organic matter in the water; Trihalomethanes, representing a by-product of chlorination; and Turbidity, measuring water clarity. Users can enter or view values for each parameter. At the bottom, a green button labeled "Predict Potability" is provided to initiate the prediction process, likely running a machine learning model to classify the water as potable (safe for drinking) or non-potable. The interface is clean, organized, and user-friendly, allowing for efficient data entry and interaction. This tool is useful for individuals or organizations needing quick insights into water quality based on these specific parameters.

## Prediction result

****

#### Result for given input data

## The image shows the result page for a "Water Quality Prediction" tool, indicating the outcome of an analysis on water potability. At the top, there is a heading labeled "Water Quality Prediction Result" in green text, setting a positive tone for the feedback. Below this, a message is prominently displayed in a white box with the text "The water is: Not Potable" in bold red font, immediately alerting the user that the water analysed does not meet the criteria for safe drinking. Beneath the result, a "Go Back" link is provided, allowing the user to return to the previous page, likely to adjust inputs or analyse another sample. At the bottom of the screen, there's a subtle note reading "Powered by Water Quality AI," which credits the AI-based too or platform responsible for the prediction.

# Chapter 11 REFERENCES

1. Reckhow, K. H. (1999). Water quality prediction and probability network models. *Canadian Journal of Fisheries and Aquatic Sciences*, *56*(7), 1150–1158. https://doi.org/10.1139/f99-040
2. S. Geetha, S. Gouthami. Internet of Things Enabled Real Time Water Quality Monitoring System ,2017
3. Azamathulla, H. M. 2013 2 – A Review on Application of Soft Computing Methods in Water Resources Engineering A2 – Yang, Xin-She. In: Metaheuristics in Water, Geotechnical and Transport Engineering (Gandomi, A. H., Talatahari, S. & Alavi, A. H., eds). Elsevier, Oxford, pp. 27–41.
4. Ashwini K, D. Diviya, J.Janice Vedha, M. Deva Priya.Intelligent Model For Predicting Water Quality.

[5] S. Geetha, S. Gouthami. Internet of Things Enabled Real Time Water Quality Monitoring System ,2017.

[6] Umair Ahmed, Rafia Mumtaz, Hirra Anwar, Asad A. Shah, Rabia Irfan and José García-Nieto. Efficient Water Quality Prediction Using Supervised Machine Learning, 2019.

[7] Ashwini K, D. Diviya, J.Janice Vedha, M. Deva Priya. Intelligent Model For Predicting Water Quality.

[8] A.N.Prasad, K. A. Mamun, F. R. Islam, H. Haqva. Smart Water Quality Monitoring System, 2015

[9] Hadi Mohammed, Ibrahim A. Hameed, Razak Seidu. Machine Learning: Based Detection of Water Contamination in Water Distribution systems,2018

[10] Priya Singh,Pankaj Deep Kaur.Review on Data Mining Techniques for Prediction of Water Quality,2017

[11] Manish Kumar Jha,Rajni Kumari Sah,M.S.Rashmitha,Rupam Sinha,B.Sujatha.Smart Water Monitoring System for Real-Time Water Quality and Usage Monitoring,2018

[12] Water Quality Monitoring System using IoT and Machine Learning, in Proceedings of the IEEE International Conference on Research in Intelligent and Computing in Engineering, pp.1-5, 2018