A Major Project Report

on

"PREDICTING URBAN WATER QUALITY WITH UBIQUITOUS DATA – A DATA-DRIVEN APPROACH"

Submitted in Partial Fulfilment of the Academic

Requirement for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering (Data Science)

Submitted By:

G.SAITEJA	(20R01A67D5)

Under the esteemed guidance of

Ms. P. Anusha

(Assistant Professor, CSE(DS) Department)

GROUP OF INSTITUTIONS
EXPLORE TO INVENT

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

(Approved by AICTE, Permanent Affiliation to JNTUH, Accredited by NBA and NAAC with 'A'Grade)
Kandlakoya, Medchal Road, Hyderabad- 501 401

https://cmrithyderabad.edu.in/

2023-24

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

(Approved by AICTE, Permanent Affiliation to JNTUH, Accredited by NBA and NAAC with 'A'Grade)
Kandlakoya, Medchal Road, Hyderabad- 501 401

CERTIFICATE

This is to certify that a Major Project entitled with: "Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach" is being submitted by:

Submitted By:

G.SAITEJA	**(20R01A67D5)**

In partial fulfillment of the requirement for award of the degree of B. Tech in CSE (Data Science) to the JNTUH, Hyderabad is a record of a bonafide work carried out under our guidance and supervision.

The results in this project have been verified and are found to be satisfactory. The results embodied in this work have not been submitted to have any other University for award of any other degree or diploma.

Signature of Guide	**Signature of Coordinator**	**Signature of HOD**
Ms. P. Anusha	**Dr. G. Bala krishna**	**Dr. A. Nirmal kumar**
(Assistant Professor)	**(Associative Professor)**	**(Associative Professor)**

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. M. Janga Reddy**, Director, **Dr. B. Satyanarayana**, Principal and **Dr. A. Nirmal kumar**, Head of Department Computer Science and Engineering (Data Science), CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to **Dr. G. Bala krishna,** Major Project Coordinator and Internal Guide **Ms. P. Anusha**, Assistant Professor, Department of Computer Science and Engineering (Data Science), CMR Institute of Technology for their constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We express our thanks to all staff members and friends for all the help and coordination extended in bringing out this Project successfully in time.

Finally, We are very much thankful to our parents and relatives who guided directly or indirectly for every step towards success.

G.SAITEJA **(20R01A67D5)**

ABSTRACT

Urban water quality is of great importance to our daily lives. Prediction of urban water quality help control water pollution and protect human health. However, predicting the urban water quality is a challenging task since the water quality varies in urban spaces non-linearly and depends on multiple factors, such as meteorology, water usage patterns, and land uses. In this work, we forecast the water quality of a station over the next few hours from a data-driven perspective, using the water quality data and water hydraulic data reported by existing monitor stations and a variety of data sources we observed in the city, such as meteorology, pipe networks, structure of road networks, and point of interests (POIs). First, we identify the influential factors that affect the urban water quality via extensive experiments. Second, we present a multi-task multi-view learning method to fuse those multiple datasets from different domains into an unified learning model. We evaluate our method with real-world datasets, and the extensive experiments verify the advantages of our method over other baselines and demonstrate the effectiveness of our approach.

INDEX

LIST OF FIGURES

1. INTRODUCTION

Urban water quality management is a pressing concern in the face of rapid urbanization and environmental degradation. As cities expand, ensuring the safety and sustainability of water resources becomes increasingly complex. Traditional approaches to monitoring and managing urban water quality are often limited in scope and effectiveness, relying on sporadic sampling and manual analysis.

In response to these challenges, our project, "Predicting Urban Water Quality With Ubiquitous Data – A Data-Driven Approach," proposes a transformative solution that harnesses the power of data analytics and ubiquitous data sources. By integrating diverse data streams, including environmental sensors, satellite imagery, social media, and citizen science initiatives, we aim to develop predictive models capable of forecasting water quality parameters in real-time.

This data-driven approach represents a paradigm shift in urban water quality management, offering several key advantages. By leveraging ubiquitous data sources, we can achieve enhanced spatial and temporal coverage, providing a more comprehensive understanding of water quality dynamics across urban landscapes. Furthermore, predictive modeling enables proactive decision-making, allowing stakeholders to anticipate changes in water quality and implement preventive measures before issues arise.

Through this project, we aspire to empower urban planners, water authorities, and policymakers with actionable insights to improve decision-making and adaptive management strategies. By leveraging the wealth of data generated in urban environments and employing advanced analytics techniques, we can pave the way towards a more sustainable and resilient urban water management system.

In summary, "Predicting Urban Water Quality With Ubiquitous Data – A Data-Driven Approach" represents an innovative endeavor to revolutionize urban water quality management. By harnessing the power of data and technology, we aim to safeguard water resources, protect public health, and promote sustainable development in cities worldwide.

1.2 OBJECTIVES

The main objective of the project is,

- Gather and integrate diverse datasets including water quality sensors, weather stations, satellite imagery, social media feeds, and municipal databases.

- Identify relevant features and variables influencing water quality, such as temperature, precipitation, land use patterns, pollutant emissions, infrastructure conditions, and socio-economic indicators.

- Develop predictive models that capture the complex relationships between different factors and water quality parameters, utilizing techniques like regression analysis, neural networks, decision trees, and ensemble methods

  .

- Validate and evaluate the performance of the predictive models through cross-validation, validation against independent datasets, and comparison with traditional monitoring methods.

- Integrate predictive models into decision support systems to assist policymakers, urban planners, and water resource managers in making informed decisions related to water quality management, pollution control, infrastructure investment, and emergency response.

1.3 PROBLEM STATEMENT

The problem statement for the project revolves around the challenge of effectively monitoring and managing water quality in urban environments amidst the complexities of diverse environmental factors, human activities, and infrastructure systems. Traditional methods of water quality monitoring often lack real-time data and spatial coverage, leading to gaps in understanding and response to water quality issues. This project aims to address these limitations by leveraging ubiquitous data sources, such as sensor networks, IoT devices, and satellite imagery, to develop predictive models that can accurately forecast urban water quality dynamics. By doing so, it seeks to empower decision-makers with timely insights and actionable information for enhancing water quality management, safeguarding public health, and promoting sustainable urban development.

2. SYSTEM PROPOSAL

2.1 EXISTING SYSTEM:

1. Traditional Monitoring Methods:

Traditional approaches to monitoring urban water quality typically rely on periodic sampling and laboratory analysis. While these methods provide valuable insights, they often suffer from limitations such as limited spatial coverage, delayed results, and high costs associated with data collection and analysis.

2. Fragmented Data Sources:

Data relevant to urban water quality is often fragmented across various sources including government agencies, research institutions, and private companies. Integration of these disparate datasets poses challenges in terms of data compatibility, consistency, and accessibility, hindering comprehensive analysis and decision-making.

3. Reactive Management Practices:

Current water quality management practices in urban areas are often reactive, responding to pollution incidents or regulatory violations after they occur. This reactive approach may lead to suboptimal outcomes in terms of pollution prevention, public health protection, and environmental sustainability.

4. Limited Predictive Capability:

The existing system lacks robust predictive capability to anticipate changes in urban water quality and proactively address emerging challenges. This limitation hampers the ability to implement preventive measures, optimize resource allocation, and mitigate the impacts of water pollution events.

5. Manual Decision-Making Processes:

Decision-making processes related to urban water quality management are predominantly manual and reliant on expert judgment. This reliance on subjective assessments may introduce biases, overlook important trends or patterns, and impede the adoption of data-driven approaches for decision support.

6. Need for a Comprehensive Solution:

There is a clear need for a comprehensive and integrated approach to predicting urban water quality, leveraging ubiquitous data sources and advanced analytics techniques. Such a solution would enable real-time monitoring, predictive modeling, and decision support capabilities to enhance water quality management practices in urban environments.

2.1.1 DISADVANTAGES:

- Limited spatial and temporal coverage due to fixed monitoring points and infrequent sampling.
- Reactive approach to addressing water quality problems, leading to potential risks and delays in corrective actions.
- High costs and resource intensiveness associated with laboratory analysis of water samples and maintaining monitoring infrastructure.

2.2 PROPOSED SYSTEM:

1. Ubiquitous Data Integration:

The proposed system will leverage ubiquitous data sources including water quality sensors, weather stations, satellite imagery, social media feeds, and municipal databases. These diverse datasets will be integrated using advanced data integration techniques to create a comprehensive and real-time understanding of urban water systems.

2. Advanced Analytics and Machine Learning:

Advanced analytics and machine learning algorithms will be employed to develop predictive models that can forecast urban water quality dynamics. Techniques such as regression analysis, neural networks, decision trees, and ensemble methods will be utilized to capture the complex relationships between different factors and water quality parameters.

3. Real-Time Monitoring and Early Warning Systems:

The system will enable real-time monitoring of key water quality indicators and the implementation of early warning systems to detect and respond to pollution events promptly. By integrating sensor data with predictive models, the system will provide actionable insights for proactive intervention and pollution prevention.

4. Decision Support and Optimization:

The proposed system will include decision support tools that assist policymakers, urban planners, and water resource managers in making informed decisions related to water quality management. These tools will optimize resource allocation, prioritize mitigation efforts, and evaluate the effectiveness of intervention strategies based on real-time data and predictive analytics.

5. Stakeholder Engagement and Collaboration:

The system will facilitate stakeholder engagement and collaboration by providing accessible and transparent information about urban water quality conditions. Stakeholders including government agencies, community organizations, and industry partners will be involved in data collection, model validation, and decision-making processes to ensure the relevance and effectiveness of the proposed system.

6. Scalability and Adaptability:

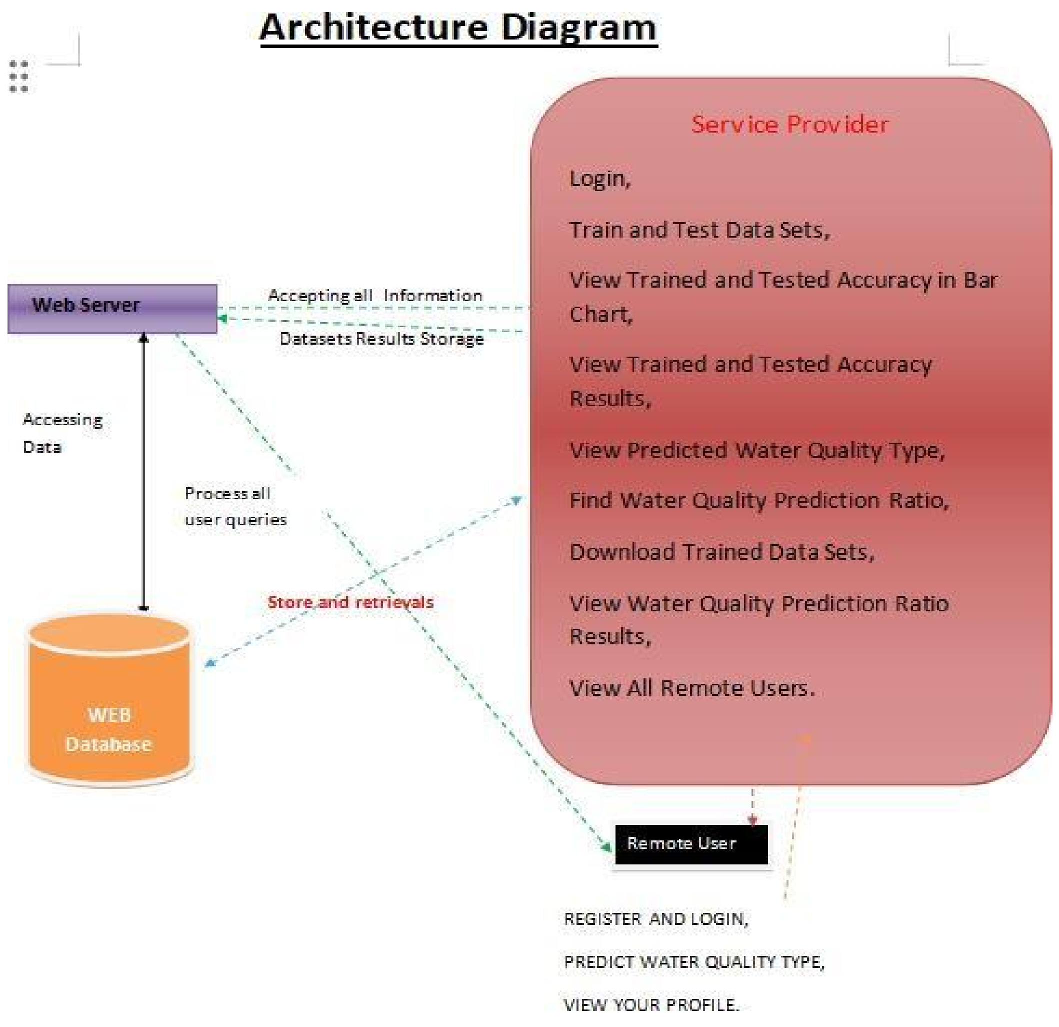
The proposed system will be designed to be scalable and adaptable to different urban contexts and evolving environmental conditions. It will be built on flexible architecture that can accommodate additional data sources, incorporate new analytical techniques, and adapt to changing regulatory requirements and stakeholder needs over time.

2.1.2 ADVANTAGES:

- Enhanced spatial and temporal coverage through the integration of diverse data sources, providing real-time insights into water quality dynamics across a broader geographical area.
- Proactive approach to water quality management, enabling stakeholders to anticipate and mitigate potential issues before they escalate, thereby reducing risks to public health and the environment.
- Cost-effectiveness by leveraging ubiquitous data sources and advanced analytics techniques, reducing reliance on expensive laboratory analysis and infrastructure maintenance.

3. SYSTEM DIAGRAMS

3.1 SYSTEM ARCHITECTURE



Architecture Diagram
Service Provider
Login,
Train and Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Predicted Water Quality Type,
Find Water Quality Prediction Ratio,
Download Trained Data Sets,
View Water Quality Prediction Ratio Results,
View All Remote Users.
Web Server
Accepting all Information
Datasets Results Storage
Accessing Data
Process all user queries
Store and retrievals
WEB Database
Remote User
REGISTER AND LOGIN,
PREDICT WATER QUALITY TYPE,
VIEW YOUR PROFILE.

Fig. 3.1: SYSTEM ARCHITECTURE

3.2 FLOW DIAGRAMS

3.2.1 FLOW CHART: Remote User

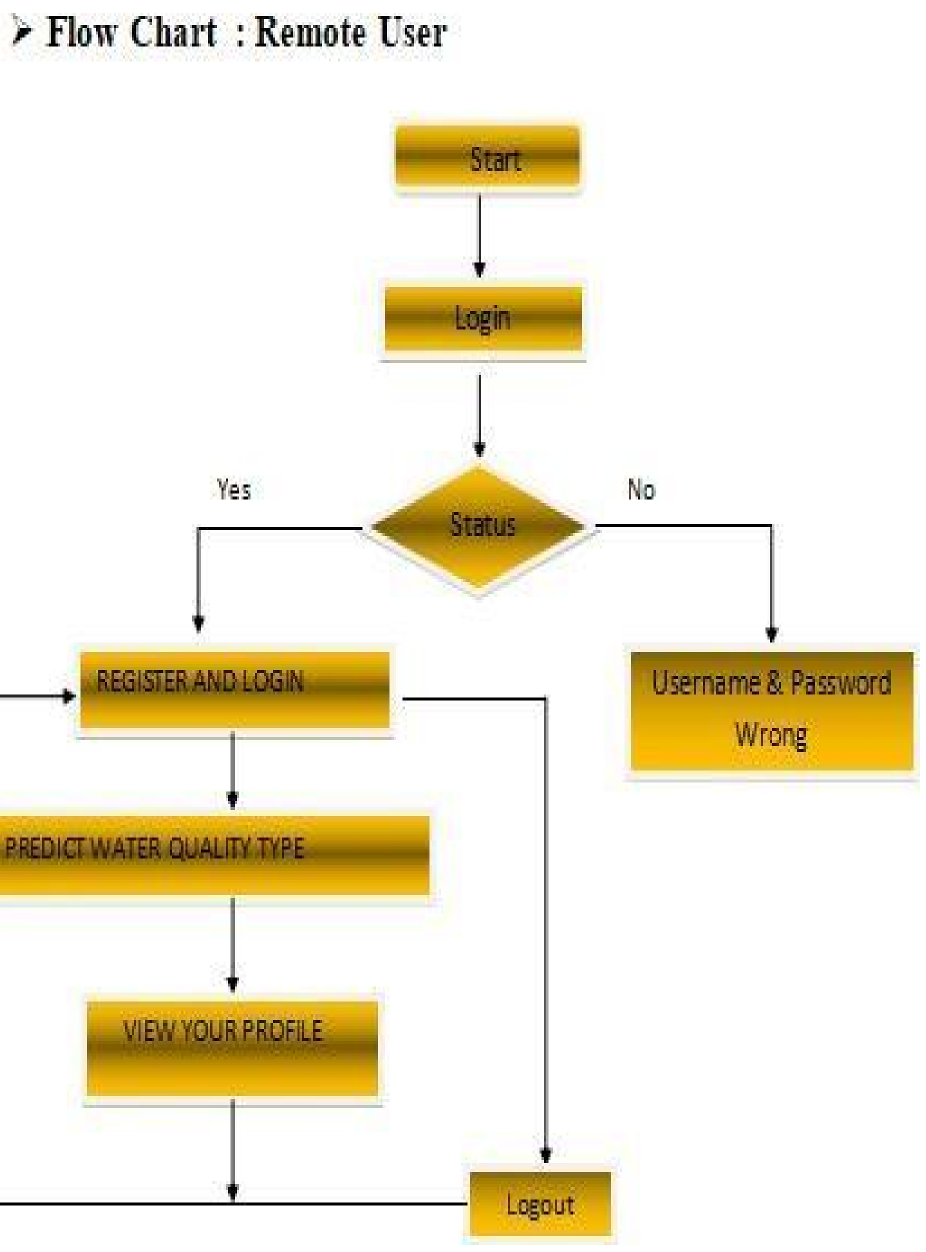
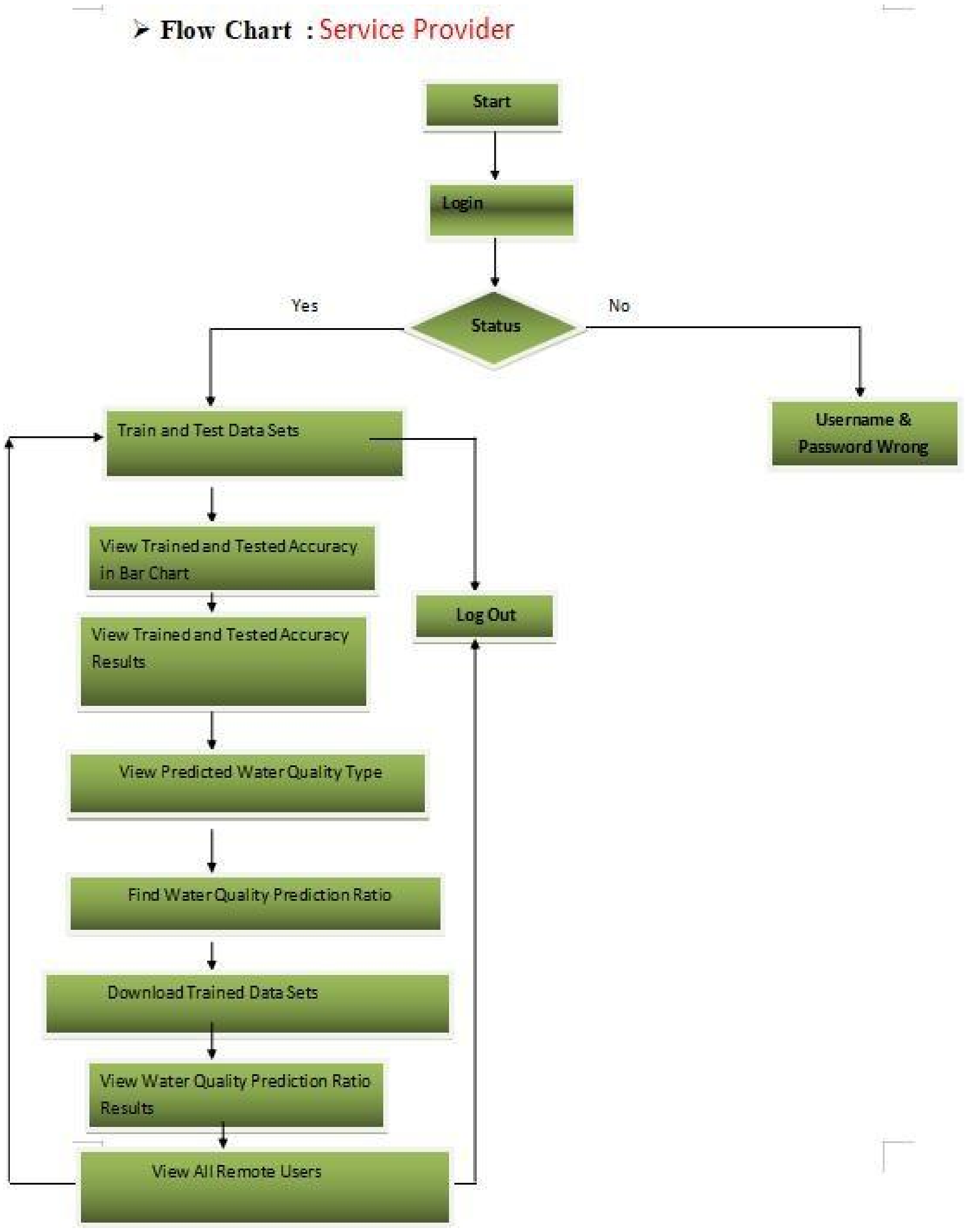

➤ Flow Chart : Remote User
Start
Login
Status
Yes
No
REGISTER AND LOGIN
Username & Password Wrong
PREDICT WATER QUALITY TYPE
VIEW YOUR PROFILE
Logout

Fig. 3.2.1 Flow Chart of Remote User

3.2.2 FLOW CHART: Service Provider



Flow Chart : Service Provider
Start
Login
Yes
Status
No
Train and Test Data Sets
Username & Password Wrong
View Trained and Tested Accuracy in Bar Chart
Log Out
View Trained and Tested Accuracy Results
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results
View All Remote Users

Figure. 3.2.2: Flow Chart of Service Provider

3.2.3 DATA FLOW DIAGRAM:

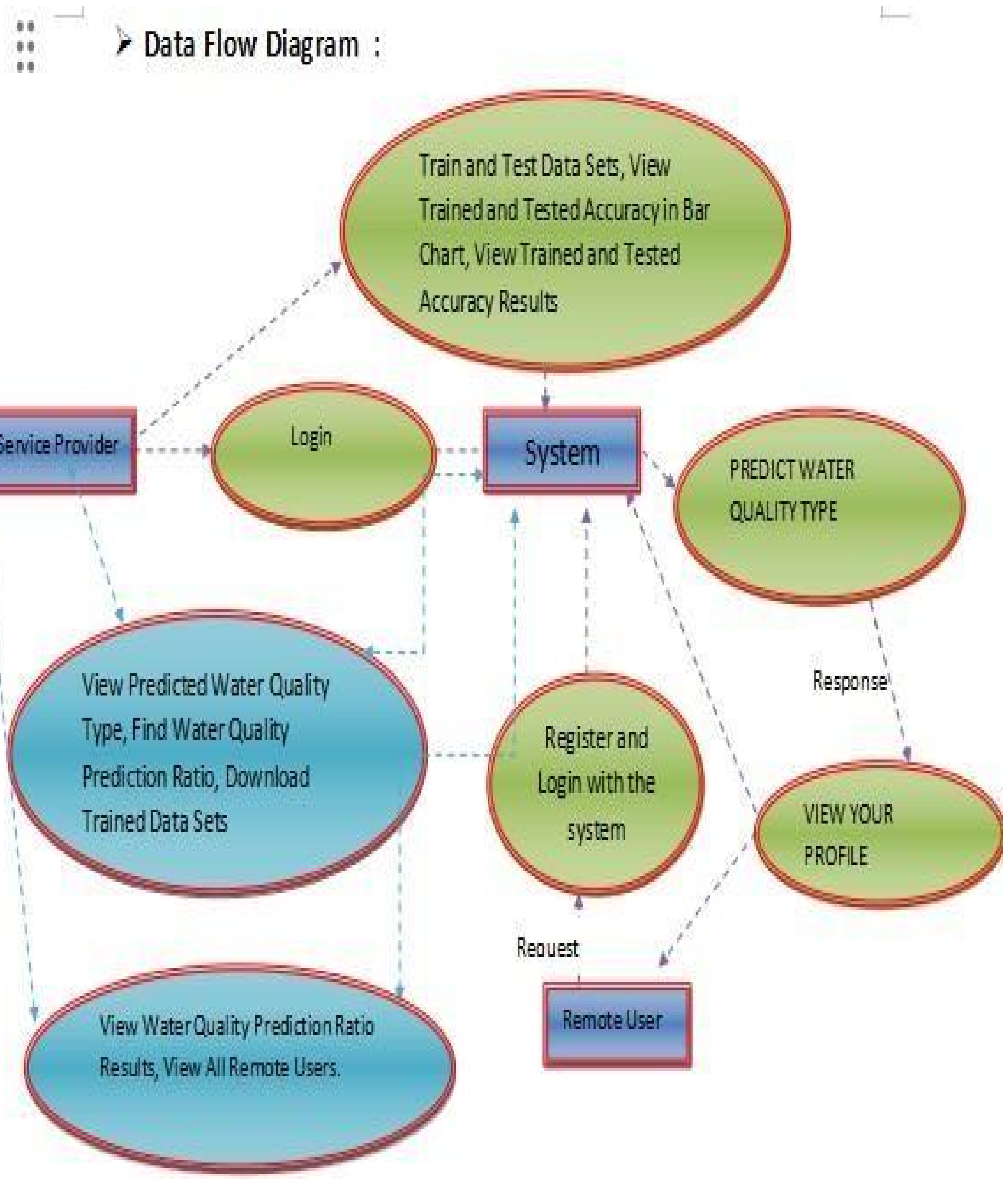

Data Flow Diagram :
Train and Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results
Service Provider
Login
System
PREDICT WATER QUALITY TYPE
Response
View Predicted Water Quality Type, Find Water Quality Prediction Ratio, Download Trained Data Sets
Register and Login with the system
VIEW YOUR PROFILE
Request
View Water Quality Prediction Ratio Results, View All Remote Users.
Remote User

Figure 3.2.3 Data Flow Diagram

3.3 UML DIAGRAMS

3.3.1 SEQUENCE DIAGRAM

➢ **Sequence Diagram**

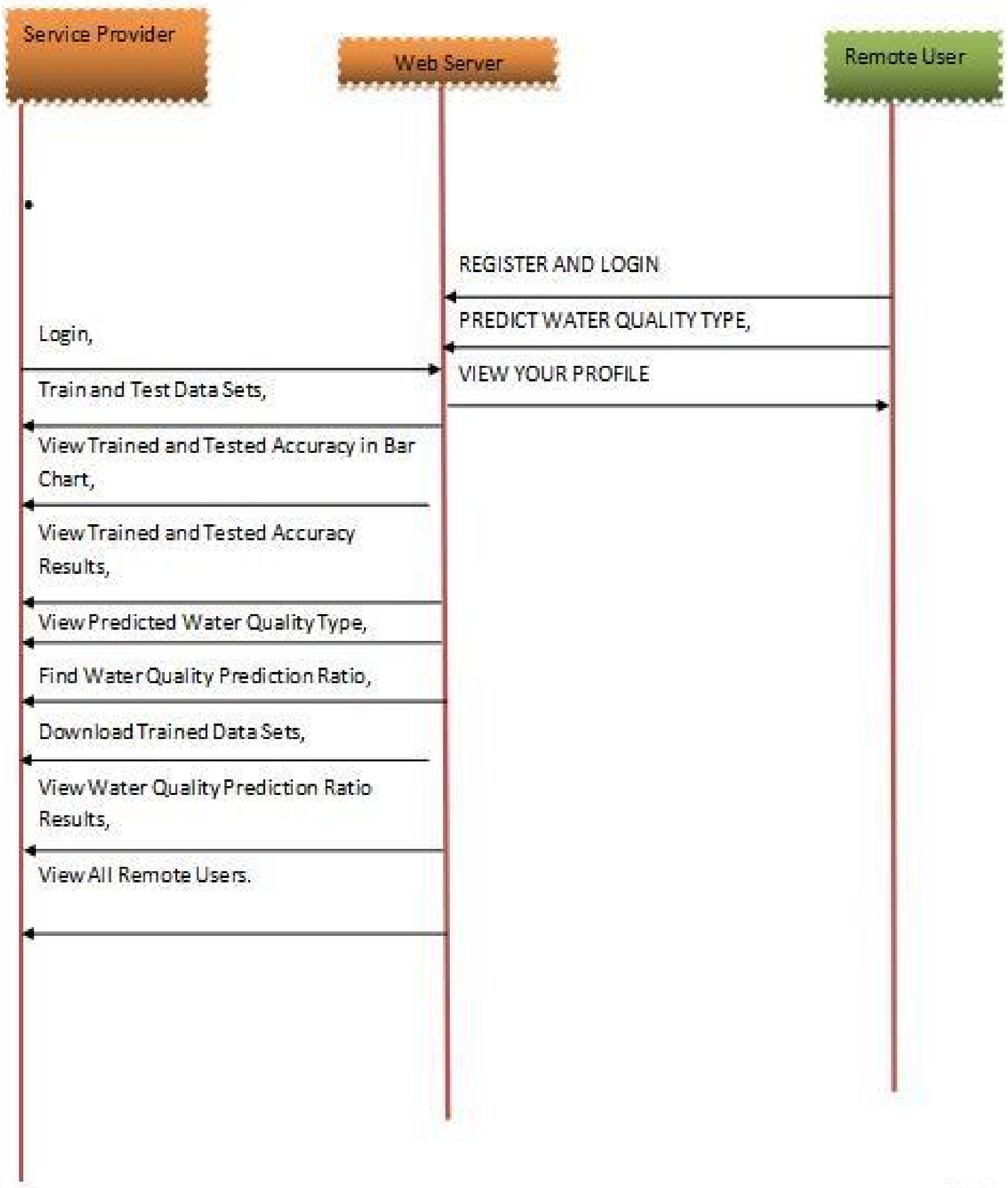
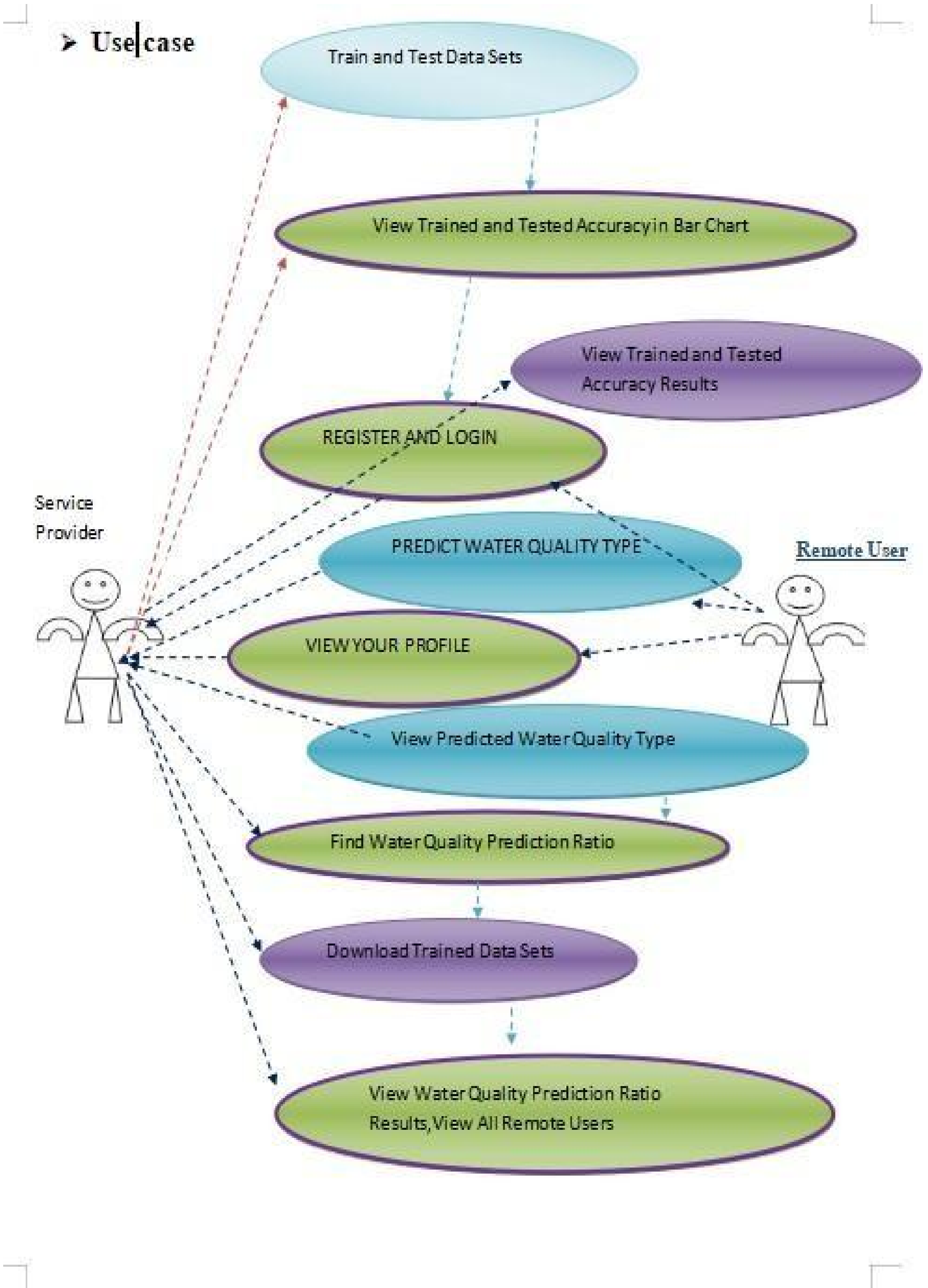

Service Provider
Web Server
Remote User
REGISTER AND LOGIN
PREDICT WATER QUALITY TYPE,
VIEW YOUR PROFILE
Login,
Train and Test Data Sets,
View Trained and Tested Accuracy in Bar Chart,
View Trained and Tested Accuracy Results,
View Predicted Water Quality Type,
Find Water Quality Prediction Ratio,
Download Trained Data Sets,
View Water Quality Prediction Ratio Results,
View All Remote Users.

Figure. 3.3.1 Sequence Diagram

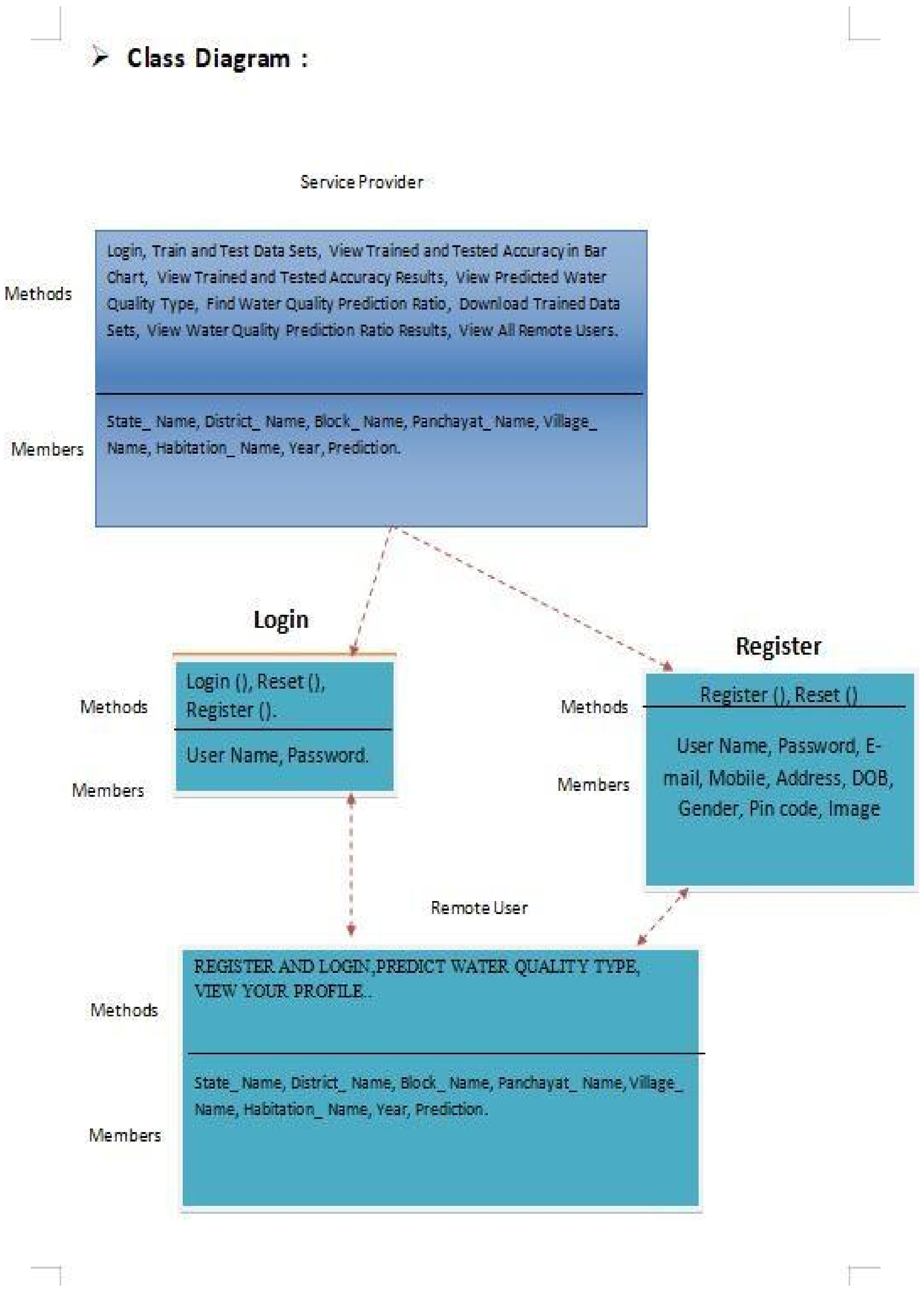
3.3.2 USE CASE DIAGRAM



➢ Use case
Train and Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
REGISTER AND LOGIN
Service Provider
PREDICT WATER QUALITY TYPE
Remote User
VIEW YOUR PROFILE
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results, View All Remote Users

Figure. 3.3.2 Use Case Diagram

3.3.3 CLASS DIAGRAM



➤ Class Diagram :
Service Provider
Methods
Login, Train and Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Predicted Water Quality Type, Find Water Quality Prediction Ratio, Download Trained Data Sets, View Water Quality Prediction Ratio Results, View All Remote Users.
Members
State_ Name, District_ Name, Block_ Name, Panchayat_ Name, Village_ Name, Habitation_ Name, Year, Prediction.
Login
Methods
Login (), Reset (), Register ().
Members
User Name, Password.
Register
Methods
Register (), Reset ()
Members
User Name, Password, E-mail, Mobile, Address, DOB, Gender, Pin code, Image
Remote User
Methods
REGISTER AND LOGIN,PREDICT WATER QUALITY TYPE, VIEW YOUR PROFILE..
Members
State_ Name, District_ Name, Block_ Name, Panchayat_ Name, Village_ Name, Habitation_ Name, Year, Prediction.

Figure. 3.3.3 Class Diagram

4. IMPLEMENTATION

4.1 MODULES

- Service Provider
- View and Authorize Users
- Remote User

4.2 MODULES DESCRIPTION:

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Train and Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Predicted Water Quality Type, Find Water QualityPrediction Ratio, Download Trained Data Sets, View Water Quality Prediction Ratio Results, View All Remote Users.

View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users.

Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT WATER QUALITY TYPE, VIEW YOUR PROFILE.

4.3 ALGORITHMS

1. Decision tree classifiers

Decision tree classifiers are used successfully in many diverse areas. Their most important feature is the capability of capturing descriptive decision making knowledge from the supplied data. Decision tree can be generated from training sets. The procedure for such generation based on the set of objects (S), each belonging to one of the classes C1, C2, …, Ck is as follows:

Step 1. If all the objects in S belong to the same class, for example Ci, the decision tree for S consists of a leaf labeled with this class

Step 2. Otherwise, let T be some test with possible outcomes O1, O2,…, On. Each object in S has one outcome for T so the test partitions S into subsets S1, S2,… Sn where each object in Si has outcome Oi for T. T becomes the root of the decision tree and for each outcome Oi we build a subsidiary decision tree by invoking the same procedure recursively on the set Si.

2. Gradient boosting

Gradient boosting is a machine learning technique used in regression and classification tasks, among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.[1][2] When a decision tree is the weak learner, the resulting algorithm is called gradient-boosted trees; it usually outperforms random forest.A gradient-boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function.

3. K-Nearest Neighbors (KNN)

- Simple, but a very powerful classification algorithm
- Classifies based on a similarity measure
- Non-parametric
- Lazy learning
- Does not "learn" until the test example is given
- Whenever we have a new data to classify, we find its K-nearest neighbors from the training data

Example:

- Training dataset consists of k-closest examples in feature space.
- Feature space means, space with categorization variables (non-metric variables).
- Learning based on instances, and thus also works lazily because instance close to the input vector for test or prediction may take time to occur in the training dataset.

4. Logistic regression Classifiers

Logistic regression analysis studies the association between a categorical dependent variable and a set of independent (explanatory) variables. The name logistic regression is used when the dependent variable has only two values, such as 0 and 1 or Yes and No. The name multinomial logistic regression is usually reserved for the case when the dependent variable has three or more unique values, such as Married, Single, Divorced, or Widowed. Although the type of data used for the dependent variable is different from that of multiple regression, the practical use of the procedure is similar.

Logistic regression competes with discriminant analysis as a method for analyzing categorical-response variables. Many statisticians feel that logistic regression is more versatile and better suited for modeling most situations than is discriminant analysis. This is because logistic regression does not assume that the independent variables are normally distributed, as discriminant analysis does.

This program computes binary logistic regression and multinomial logistic regression on both numeric and categorical independent variables. It reports on the regression equation as well as the goodness of fit, odds ratios, confidence limits, likelihood, and deviance. It performs a comprehensive residual analysis including diagnostic residual reports and plots. It can perform an independent variable subset selection search, looking for the best regression model with the fewest independent variables. It provides confidence intervals on predicted values and provides ROC curves to help determine the best cutoff point for classification. It allows you to validate your results by automatically classifying rows that are not used during the analysis.

5. Naïve Bayes

The naive bayes approach is a supervised learning method which is based on a simplistic hypothesis: it assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature .

Yet, despite this, it appears robust and efficient. Its performance is comparable to other supervised learning techniques. Various reasons have been advanced in the literature. In this tutorial, we highlight an explanation based on the representation bias. The naive bayes classifier is a linear classifier, as well as linear discriminant analysis, logistic regression or linear SVM (support vector machine). The difference lies on the method of estimating the parameters of the classifier (the learning bias).

While the Naive Bayes classifier is widely used in the research world, it is not widespread among practitioners which want to obtain usable results. On the one hand, the researchers found especially it is very easy to program and implement it, its parameters are easy to estimate, learning is very fast even on very large databases, its accuracy is reasonably good in comparison to the other approaches. On the other hand, the final users do not obtain a model easy to interpret and deploy, they does not understand the interest of such a technique.

6. Random Forest

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

The first algorithm for random decision forests was created in 1995 by Tin Kam Ho[1] using the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

Random forests are frequently used as "blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

7. SVM

In classification tasks a discriminant machine learning technique aims at finding, based on an independent and identically distributed (iid) training dataset, a discriminant function that can correctly predict labels for newly acquired instances. Unlike generative machine learning approaches, which require computations of conditional probability distributions, a discriminant classification function takes a data point x and assigns it to one of the different classes that are a part of the classification task. Less powerful than generative approaches, which are mostly used when prediction involves outlier detection, discriminant approaches require fewer computational resources and less training data, especially for a multidimensional feature space and when only posterior probabilities are needed. is equivalent to finding the equation for a multidimensional surface that best separates the different classes in the feature space.

SVM is a discriminant technique, and, because it solves the convex optimization problem analytically, it always returns the same optimal hyperplane parameter—in contrast to genetic algorithms (GAs) or perceptrons, both of which are widely used for classification in machine learning. For perceptrons, solutions are highly dependent on the initialization and termination criteria. For a specific kernel that transforms the data from the input space to the feature space, training returns uniquely defined SVM model parameters for a given training set, whereas the perceptron and GA classifier models are different each time training is initialized.

5.SYSTEM REQUIREMENTS

5.1 HARDWARE REQUIREMENTS

➢ Processor	-	Pentium –IV
➢ RAM	-	4 GB (min)
➢ Hard Disk	-	20 GB
➢ Key Board	-	Standard Windows Keyboard
➢ Mouse	-	Two or Three Button Mouse
➢ Monitor	-	SVGA

5.2 SOFTWARE REQUIREMENTS

➢ Operating system	:	Windows 7 Ultimate.
➢ Coding Language	:	Python.
➢ Front-End	:	Python.
➢ Back-End	:	Django
➢ Designing	:	HTML, CSS, Java Script.
➢ Data Base	:	MySQL (WAMP Server).

5.3 SYSTEM STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity.

5.4 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.4.1 TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.
Systems/Procedures:	interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specifica

tion or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

5.4.2 TESTING METHODOLOGIES

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

OTHER TESTING METHODOLOGIES:

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package “Virtual Private Network” has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user’s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

6. SOURCE CODE

Remote User:

Views.py:

```
from django.shortcuts import render, redirect, get_object_or_404

import pandas          as pd

from sklearn.metrics import classification_report,accuracy_score,confusion_matrix

# Create your views here.

From Remote_User.models import
ClientRegister_Model,water_quality_type,detection_ratio,detection_accuracy

def login(request):

     if request.method == "POST" and 'submit1' in request.POST:

        username = request.POST.get('username')

        password  =  request.POST.get('password')

        try:

           enter = ClientRegister_Model.objects.get(username=username,password=password)

           request.session["userid"] = enter.id


           return redirect('ViewYourProfile')

        except:

           pass

    return render(request,'RUser/login.html')

def Register1(request):

    if request.method == "POST":

        username = request.POST.get('username')

        email = request.POST.get('email')

        password = request.POST.get('password')

        phoneno = request.POST.get('phoneno')

        country = request.POST.get('country')

        state = request.POST.get('state')
        city = request.POST.get('city')
        address = request.POST.get('address')
        gender = request.POST.get('gender')
        ClientRegister_Model.objects.create(username=username, email=email, password=password,
    phoneno=phoneno,        country=country, state=state, city=city,address=address, gender=gender)
        obj = "Registered Successfully"
        return render(request, 'RUser/Register1.html',{'object':obj})
    else:
        return render(request,'RUser/Register1.html')
def ViewYourProfile(request):
    userid = request.session['userid']
    obj = ClientRegister_Model.objects.get(id= userid)
    return render(request,'RUser/ViewYourProfile.html',{'object':obj})
def Predict_Water_Quality(request):
    if request.method == "POST":
        if request.method == "POST":
            State_Name = request.POST.get('State_Name')
            District_Name = request.POST.get('District_Name')
            Block_Name = request.POST.get('Block_Name')
            Panchayat_Name = request.POST.get('Panchayat_Name')
            Village_Name = request.POST.get('Village_Name')
            Habitation_Name = request.POST.get('Habitation_Name')
            Year = request.POST.get('Year')
        df = pd.read_csv('Water_Quality_Datasets.csv', encoding='latin-1')
        def apply_results(results):
            if (results == 'Salinity'):
                return 0
            elif (results == 'Fluoride'):
                return 1
    elif (results == 'Iron'):
        return 2


    elif (results == 'Arsenic'):return 3
    elif (results == 'Nitrate'):
        return 4
df['results'] = df['Quality_Parameter'].apply(apply_results)
X = df['Habitation_Name']
y = df['results']
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
x = cv.fit_transform(X)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape
print("Naive Bayes")
from sklearn.naive_bayes import
 MultinomialNBNB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print("ACCURACY")
print(naivebayes)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_nb))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_nb))
detection_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)
```

```
models.append(('naive_bayes', NB))


# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print("ACCURACY")
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
models.append(('SVM', lin_clf))
print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) * 100)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))
detection_accuracy.objects.create(names="LogisticRegression",
ratio=accuracy_score(y_test, y_pred) * 100)
```

```
        models.append(('LogisticRegression', reg))

        from sklearn.ensemble import VotingClassifier

        classifier = VotingClassifier(models)

    classifier.fit(X_train, y_train)

        y_pred = classifier.predict(X_test)

        Habitation_Name = [Habitation_Name]

        vector1 = cv.transform(Habitation_Name).toarray()

        predict_text = classifier.predict(vector1)

        pred = str(predict_text).replace("[", "")

        pred1 = str(pred.replace("]", ""))

        prediction = int(pred1)

        if (prediction == 0):

            val = 'Salinity'

        elif (prediction == 1):

            val = 'Fluoride'

        elif (prediction == 2):

            val = 'Iron'

        elif (prediction == 3):

            val = 'Arsenic-Fully Polluted'

        elif (prediction == 4):

            val = 'Nitrate'

            print(prediction)

            print(val)

water_quality_type.objects.create(State_Name=State_Name,District_Name=District_Name,B
lock_Name=Block_Name,Panchayat_Name=Panchayat_Name,Village_Name=Village_Name
,Habitation_Name=Habitation_Name,Year=Year,Prediction=val)

        return render(request, 'RUser/Predict_Water_Quality.html',{'objs': val})

    return render(request, 'RUser/Predict_Water_Quality.html'
```

Service_Provider:

Views.py:

```
from django.db.models import Count, Avg

from django.shortcuts import render, redirect

from django.db.models import Count

from django.db.models import Q

import datetime

import xlwt

from django.http import HttpResponse

import pandas as pd

import warnings

warnings.filterwarnings("ignore")

from wordcloud import WordCloud,STOPWORDS

stopwords = set(STOPWORDS)

import re

from collections import Counter

from sklearn.metrics import classification_report,accuracy_score,confusion_matrix

# Create your views here.

from Remote_User.models import
ClientRegister_Model,water_quality_type,detection_ratio,detection_accuracy

def serviceproviderlogin(request):

    if request.method  == "POST":

        admin = request.POST.get('username')

        password = request.POST.get('password')

        if admin == "Admin" and password =="Admin":

            detection_accuracy.objects.all().delete()

            return redirect('View_Remote_Users')

    return render(request,'SProvider/serviceproviderlogin.html')

 def Find_Water_Quality_Predicted_Ratio(request):
```

```
detection_ratio.objects.all().delete()
ratio = ""
kword = 'Salinity'
print(kword)
obj = water_quality_type.objects.all().filter(Q(Prediction=kword))
obj1 = water_quality_type.objects.all()
count = obj.count();
count1 = obj1.count();
ratio = (count / count1) * 100
if ratio != 0:
   detection_ratio.objects.create(names=kword, ratio=ratio)
ratio1 = ""
kword1 = 'Fluoride'
print(kword1)
obj1 = water_quality_type.objects.all().filter(Q(Prediction=kword1))
obj11 = water_quality_type.objects.all()
count1 = obj1.count();
count11 = obj11.count();
ratio1 = (count1 / count11) * 100
if ratio1 != 0:
   detection_ratio.objects.create(names=kword1, ratio=ratio1)
ratio12 = ""
kword12 = 'Iron'
print(kword12)
obj12 = water_quality_type.objects.all().filter(Q(Prediction=kword12))
obj112 = water_quality_type.objects.all()
count12 = obj12.count();
count112 = obj112.count();
```

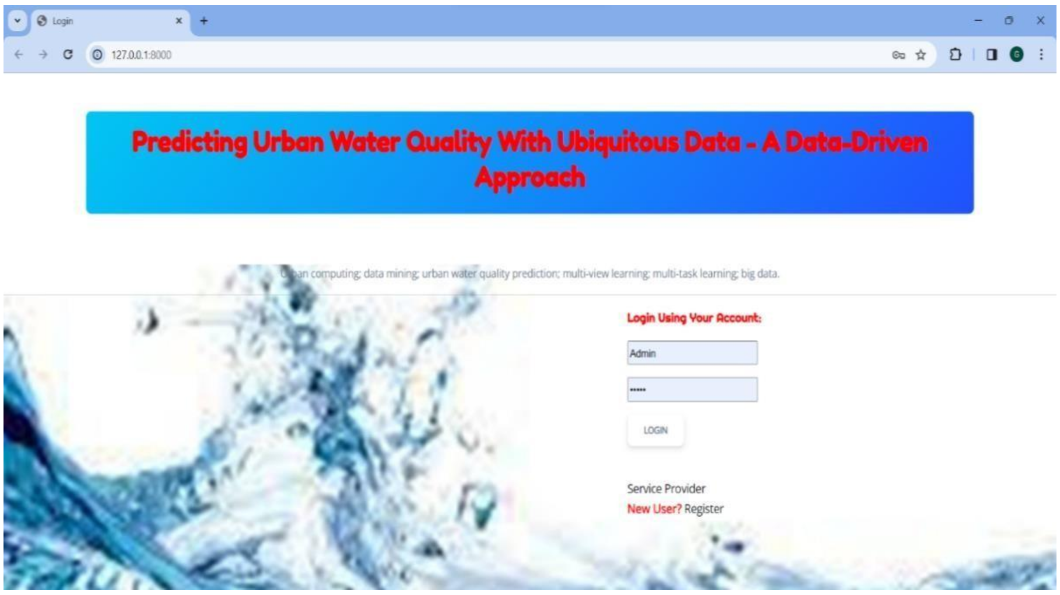
```
   ratio12 = (count12 / count112) * 100
   if ratio12 != 0:
       detection_ratio.objects.create(names=kword12, ratio=ratio12)
   ratio123 = ""
   kword123 = 'Arsenic-Fully Polluted'
   print(kword123)
   obj123 = water_quality_type.objects.all().filter(Q(Prediction=kword123))
   obj1123 = water_quality_type.objects.all()
   count123 = obj123.count();
   count1123 = obj1123.count();
   ratio123 = (count123 / count1123) * 100
   if ratio123 != 0:
       detection_ratio.objects.create(names=kword123, ratio=ratio123)
   obj = detection_ratio.objects.all()
   return render(request, 'SProvider/Find_Water_Quality_Predicted_Ratio.html', {'objs': obj})
 def View_Remote_Users(request):
   obj=ClientRegister_Model.objects.all()
   return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})
 def ViewTrendings(request):
   topic =
water_quality_type.objects.values('topics').annotate(dcount=Count('topics')).order_by('-
dcount')
   return render(request,'SProvider/ViewTrendings.html',{'objects':topic})
def charts(request,chart_type):
   chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
   return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})
def charts1(request,chart_type):
  chart1 = detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
   return render(request,"SProvider/charts1.html", {'form':chart1, 'chart_type':chart_type})
```

```
def View_Predicted_Water_Quality(request):
    obj =water_quality_type.objects.all()
    return render(request, 'SProvider/View_Predicted_Water_Quality.html', {'list_objects':
obj})
def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})
def Download_Trained_DataSets(request):
   response = HttpResponse(content_type='application/ms-excel')
    # decide file name
    response['Content-Disposition'] = 'attachment; filename="PredictedData.xls"'
    # creating workbook
    wb = xlwt.Workbook(encoding='utf-8')
    # adding sheet
    ws = wb.add_sheet("sheet1")
    # Sheet header, first row
    row_num = 0
    font_style = xlwt.XFStyle()
    # headers are bold
    font_style.font.bold = True
    # writer = csv.writer(response)
    obj = water_quality_type.objects.all()
    data = obj # dummy method to fetch data.
    for my_row in data:
        row_num = row_num + 1
        ws.write(row_num, 0, my_row.State_Name, font_style)
        ws.write(row_num, 1, my_row.District_Name, font_style)
        ws.write(row_num, 2, my_row.Block_Name, font_style)
        ws.write(row_num, 3, my_row.Panchayat_Name, font_style)
        ws.write(row_num, 4, my_row.Village_Name, font_style)
        ws.write(row_num, 5, my_row.Habitation_Name, font_style)
        ws.write(row_num, 6, my_row.Year, font_style)
        ws.write(row_num, 7, my_row.Prediction, font_style)
    wb.save(response)
    return response
def train_model(request):
    detection_accuracy.objects.all().delete()
df = pd.read_csv('Water_Quality_Datasets.csv',encoding='latin-1')
    def apply_results(results):
        if (results == 'Salinity'):
            return 0
        elif (results == 'Fluoride'):
            return 1
        elif (results == 'Iron'):
            return 2
        elif (results == 'Arsenic'):
            return 3
        elif (results == 'Nitrate'):
            return 4
    df['results'] = df['Quality_Parameter'].apply(apply_results)
    X = df['Habitation_Name']
    y = df['results']
    from sklearn.feature_extraction.text import CountVectorizer
    cv = CountVectorizer(lowercase=False, strip_accents='unicode', ngram_range=(1, 1))
    x = cv.fit_transform(X)
    models = []
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape
print("Naive Bayes")
from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb = NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) * 100
print("ACCURACY")
print(naivebayes)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_nb))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_nb))
detection_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)
models.append(('naive_bayes', NB))
# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100
print("ACCURACY")
print(svm_acc)
print("CLASSIFICATION REPORT")
print(classification_report(y_test, predict_svm))
```

```
 print("CONFUSION MATRIX")
 print(confusion_matrix(y_test, predict_svm))
 detection_accuracy.objects.create(names="SVM", ratio=svm_acc)
 models.append(('SVM', lin_clf))
 print("Logistic Regression")
 from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train, y_train)
 y_pred = reg.predict(X_test)
 print("ACCURACY")
 print(accuracy_score(y_test, y_pred) * 100)
 print("CLASSIFICATION REPORT")
 print(classification_report(y_test, y_pred))
 print("CONFUSION MATRIX")
 print(confusion_matrix(y_test, y_pred))
 detection_accuracy.objects.create(names="LogisticRegression",
 ratio=accuracy_score(y_test, y_pred) * 100)
 models.append(('LogisticRegression', reg))
 csv_format = 'Results.csv'
 df.to_csv(csv_format, index=False)
 return render(request,'SProvider/train_model.html', {'objs': obj})
```

7. RESULTS



Login

127.0.0.1:8000

Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach

Urban computing; data mining; urban water quality prediction; multi-view learning; multi-task learning; big data.

Login Using Your Account:

Admin

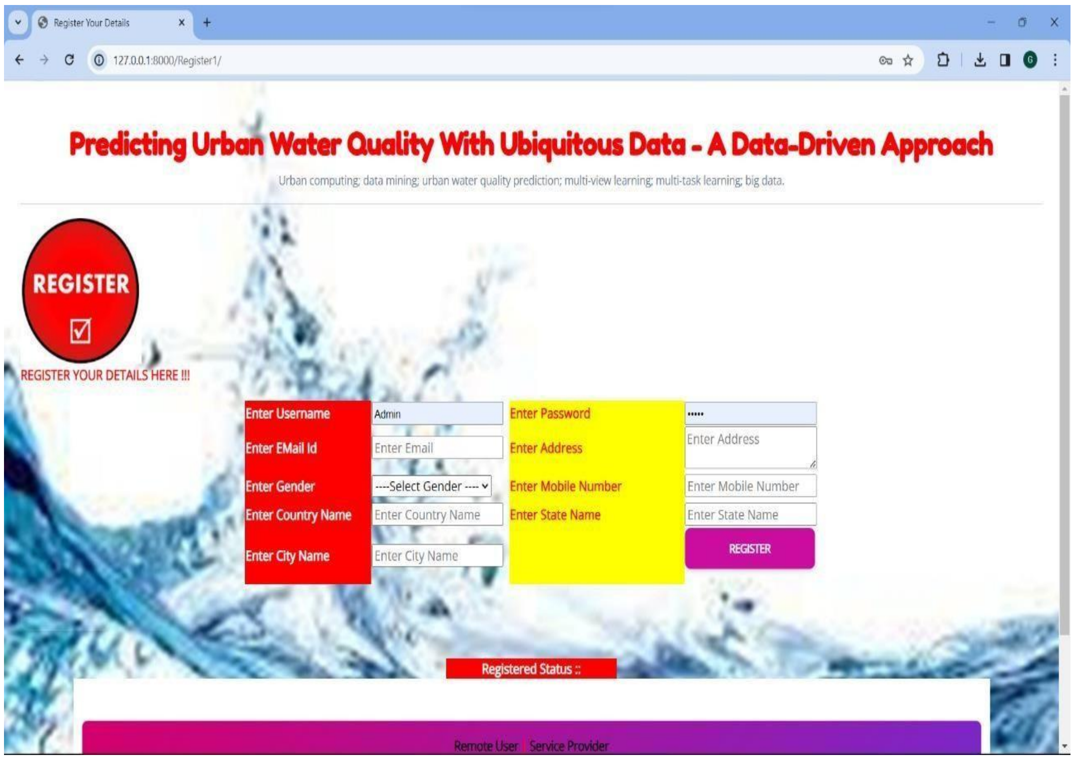
•••••

LOGIN

Service Provider

New User? Register

Figure 7.1 : HOME PAGE



Register Your Details

127.0.0.1:8000/Register1/

Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach

Urban computing; data mining; urban water quality prediction; multi-view learning; multi-task learning; big data.

REGISTER

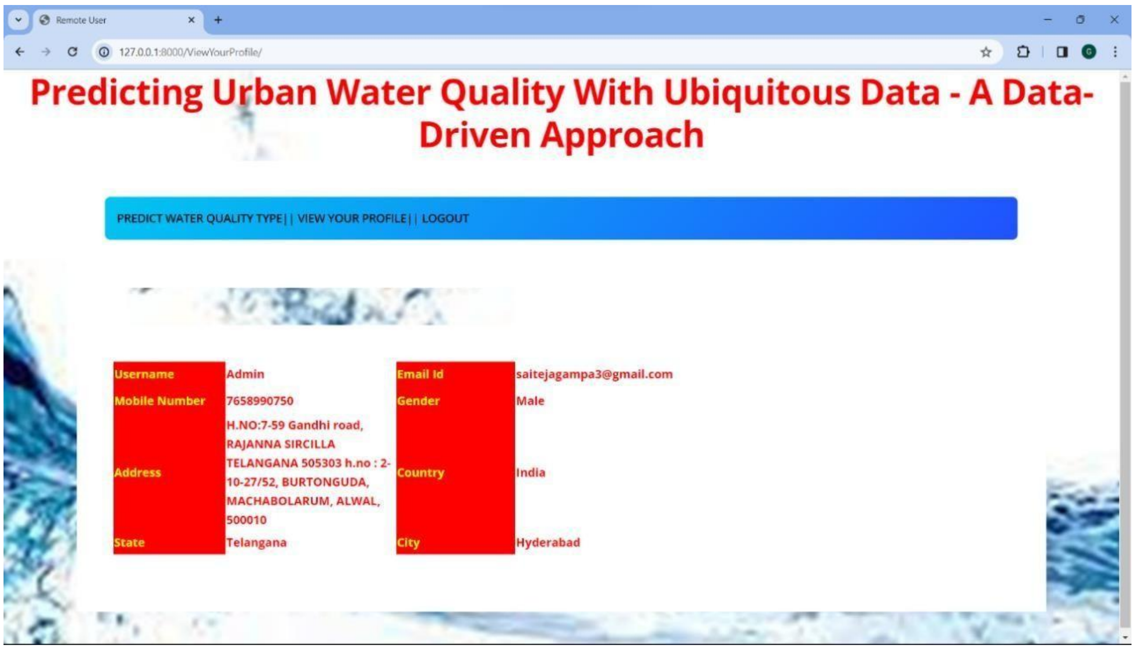
REGISTER YOUR DETAILS HERE !!!

Enter Username	Admin	Enter Password	•••••
Enter EMail Id	Enter Email	Enter Address	Enter Address
Enter Gender	----Select Gender ----	Enter Mobile Number	Enter Mobile Number
Enter Country Name	Enter Country Name	Enter State Name	Enter State Name
Enter City Name	Enter City Name		REGISTER

Registered Status ::

Remote User Service Provider

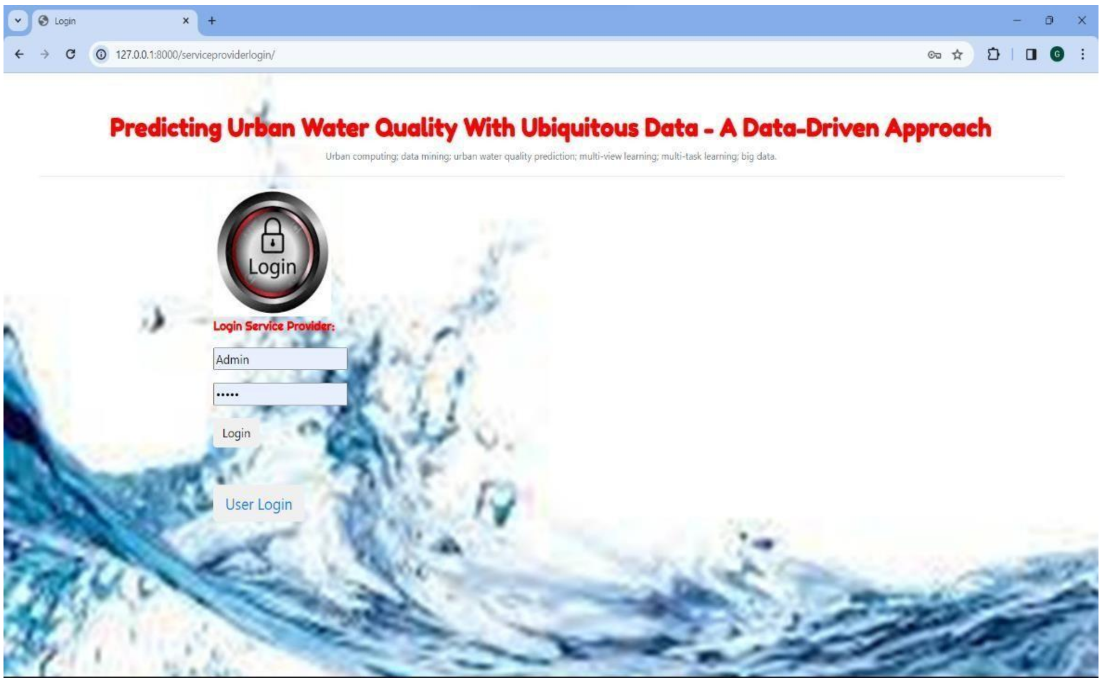
Figure 7.2 : NEW CANDIDATE REGISTRATION



Remote User
127.0.0.1:8000/ViewYourProfile/
Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach
PREDICT WATER QUALITY TYPE| | VIEW YOUR PROFILE| | LOGOUT

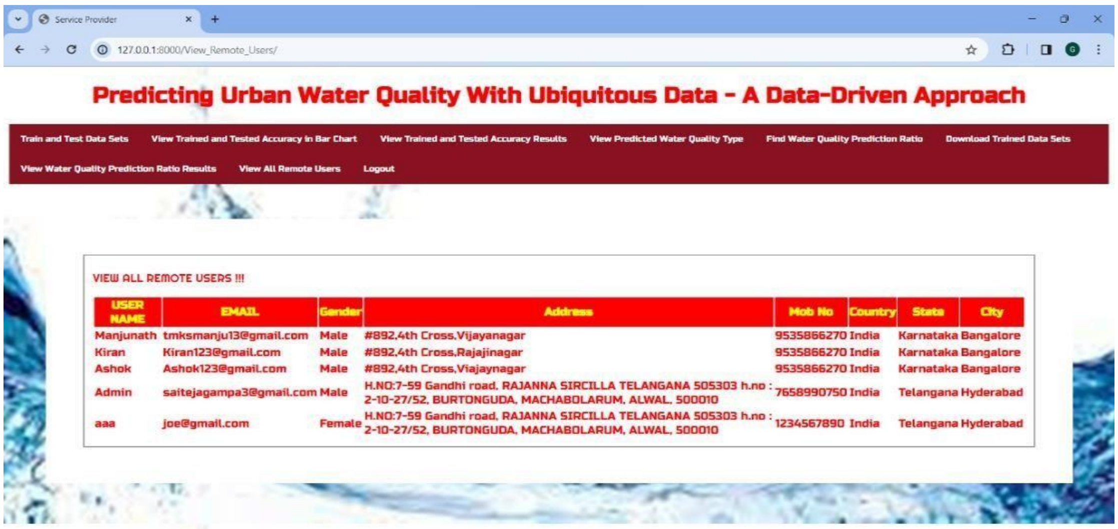
Username	Admin	Email Id	saitejagampa3@gmail.com
Mobile Number	7658990750	Gender	Male
Address	H.NO:7-59 Gandhi road, RAJANNA SIRCILLA TELANGANA 505303 h.no : 2-10-27/52, BURTONGUDA, MACHABOLARUM, ALWAL, 500010	Country	India
State	Telangana	City	Hyderabad

Figure.7.3 PROFILE DETAILS



Login
127.0.0.1:8000/serviceproviderlogin/
Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach
Urban computing; data mining; urban water quality prediction; multi-view learning; multi-task learning; big data.
Login
Login Service Provider:
Admin
•••••
Login
User Login

Figure.7.4 MOVE TO SERVICE PROVIDER AND DO LOGIN


Service Provider
127.0.0.1:8000/View_Remote_Users/
Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach
Train and Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results
View All Remote Users
Logout

VIEW ALL REMOTE USERS !!!

USER NAME	EMAIL	Gender	Address	Mob No	Country	State	City
Manjunath	tmksmanju13@gmail.com	Male	#892,4th Cross,Vijayanagar	9535866270	India	Karnataka	Bangalore
Kiran	Kiran123@gmail.com	Male	#892,4th Cross,Rajajinagar	9535866270	India	Karnataka	Bangalore
Ashok	Ashok123@gmail.com	Male	#892,4th Cross,Viajaynagar	9535866270	India	Karnataka	Bangalore
Admin	saitejagampa3@gmail.com	Male	H.NO:7-59 Gandhi road, RAJANNA SIRCILLA TELANGANA 505303 h.no : 2-10-27/52, BURTONGUDA, MACHABOLARUM, ALWAL, 500010	7658990750	India	Telangana	Hyderabad
aaa	joe@gmail.com	Female	H.NO:7-59 Gandhi road, RAJANNA SIRCILLA TELANGANA 505303 h.no : 2-10-27/52, BURTONGUDA, MACHABOLARUM, ALWAL, 500010	1234567890	India	Telangana	Hyderabad

Figure 7.5 REMOTE USER'S ADDED TO THE SERVICE PROVIDER LIST

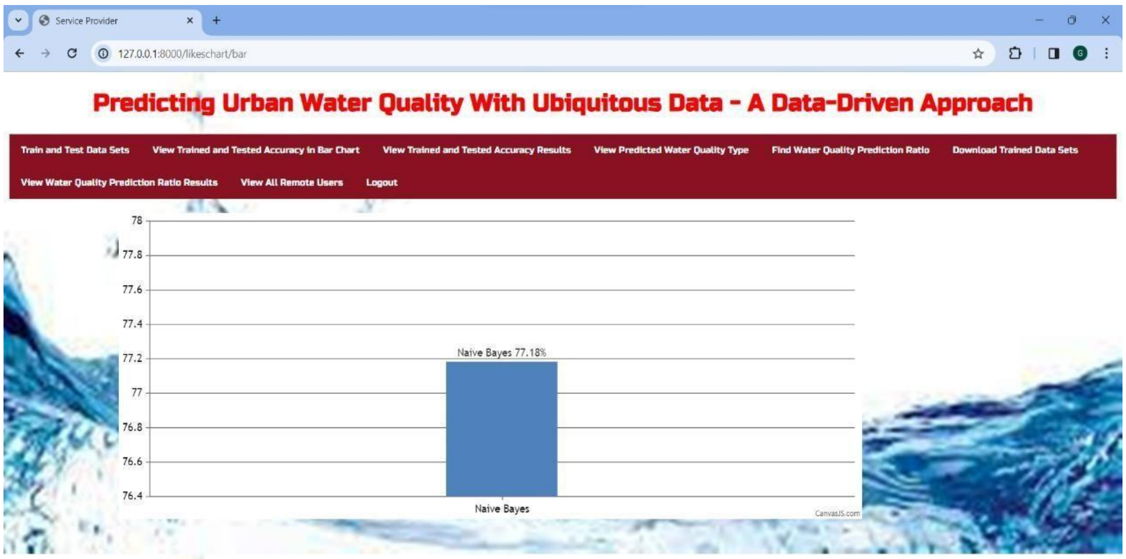
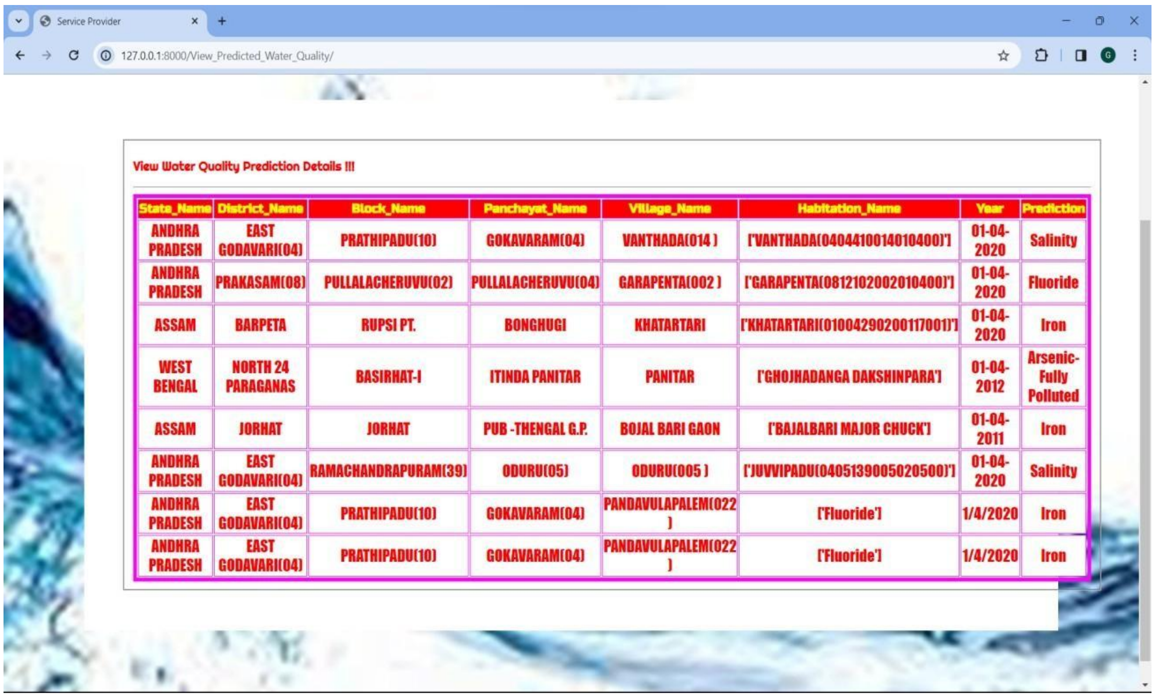

Service Provider
127.0.0.1:8000/likeschart/bar
Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach
Train and Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results
View All Remote Users
Logout
78
77.8
77.6
77.4
77.2
77
76.8
76.6
76.4
Naive Bayes 77.18%
Naive Bayes
CanvasJS.com

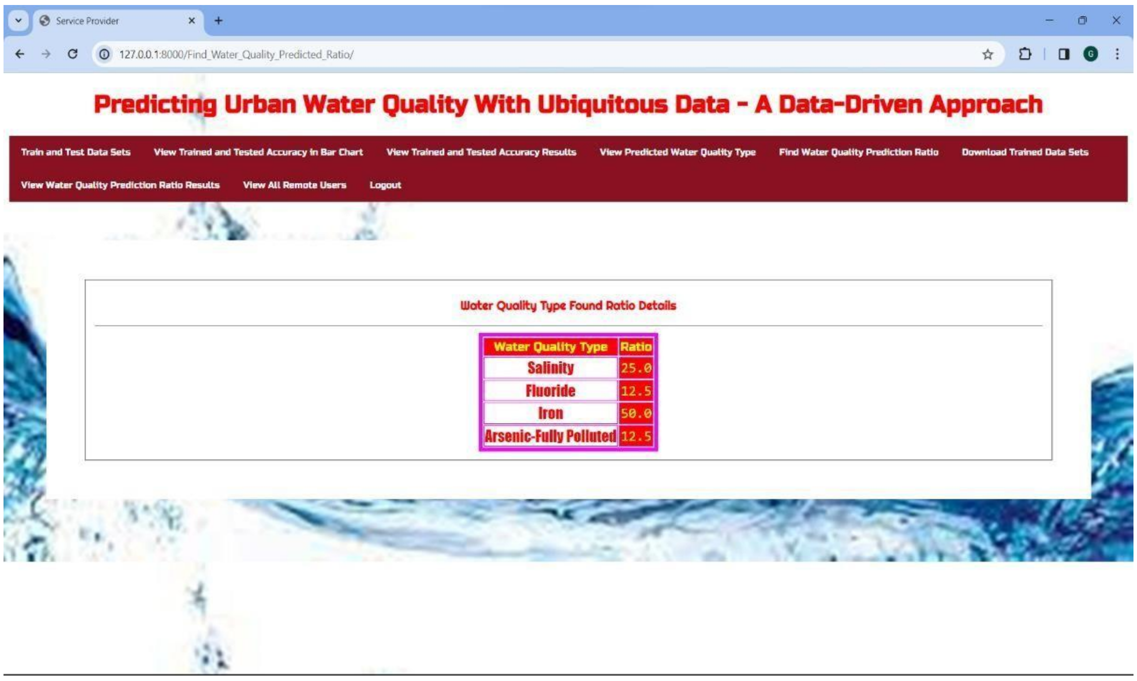
Figure 7.6 BAR GRAPH REPRESENTATION



View Water Quality Prediction Details !!!

State_Name	District_Name	Block_Name	Panchayat_Name	Village_Name	Habitation_Name	Year	Prediction
ANDHRA PRADESH	EAST GODAVARI(04)	PRATHIPADU(10)	GOKAVARAM(04)	VANTHADA(014)	['VANTHADA(04044100140104000)']	01-04-2020	Salinity
ANDHRA PRADESH	PRAKASAM(08)	PULLALACHERUVU(02)	PULLALACHERUVU(04)	GARAPENTA(002)	['GARAPENTA(08121020020104000)']	01-04-2020	Fluoride
ASSAM	BARPETA	RUPSI PT.	BONGHUGI	KHATARTARI	['KHATARTARI(01004290200117001)']	01-04-2020	Iron
WEST BENGAL	NORTH 24 PARAGANAS	BASIRHAT-I	ITINDA PANITAR	PANITAR	['GHOJHADANGA DAKSHINPARA']	01-04-2012	Arsenic-Fully Polluted
ASSAM	JORHAT	JORHAT	PUB -THENGAL G.P.	BOJAL BARI GAON	['BAJALBARI MAJOR CHUCK']	01-04-2011	Iron
ANDHRA PRADESH	EAST GODAVARI(04)	RAMACHANDRAPURAM(39)	ODURU(05)	ODURU(005)	['JUVVIPADU(04051390050205000)']	01-04-2020	Salinity
ANDHRA PRADESH	EAST GODAVARI(04)	PRATHIPADU(10)	GOKAVARAM(04)	PANDAVULAPALEM(022)	['Fluoride']	1/4/2020	Iron
ANDHRA PRADESH	EAST GODAVARI(04)	PRATHIPADU(10)	GOKAVARAM(04)	PANDAVULAPALEM(022)	['Fluoride']	1/4/2020	Iron

Figure 7.7 WATER QUALITY PREDICTION DETAILS


Service Provider
127.0.0.1:8000/Find_Water_Quality_Predicted_Ratio/
Train and Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results
View All Remote Users
Logout

Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach

Water Quality Type Found Ratio Details

Water Quality Type	Ratio
Salinity	25.0
Fluoride	12.5
Iron	50.0
Arsenic-Fully Polluted	12.5

Figure 7.8 WATER QUALITY TYPE RATIOS

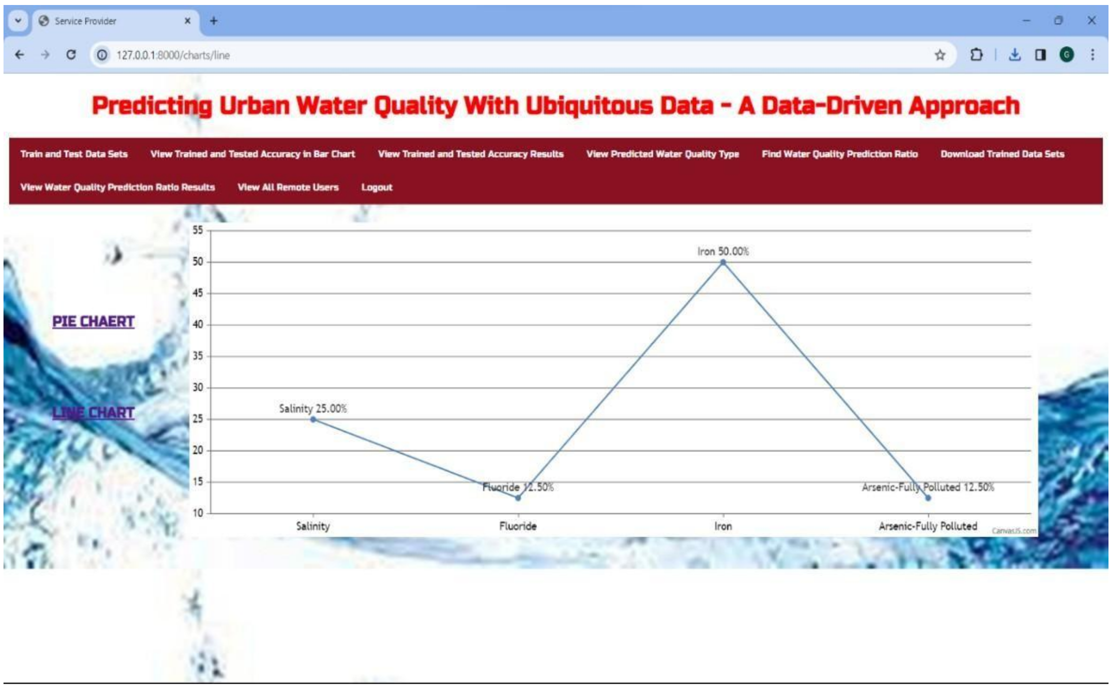
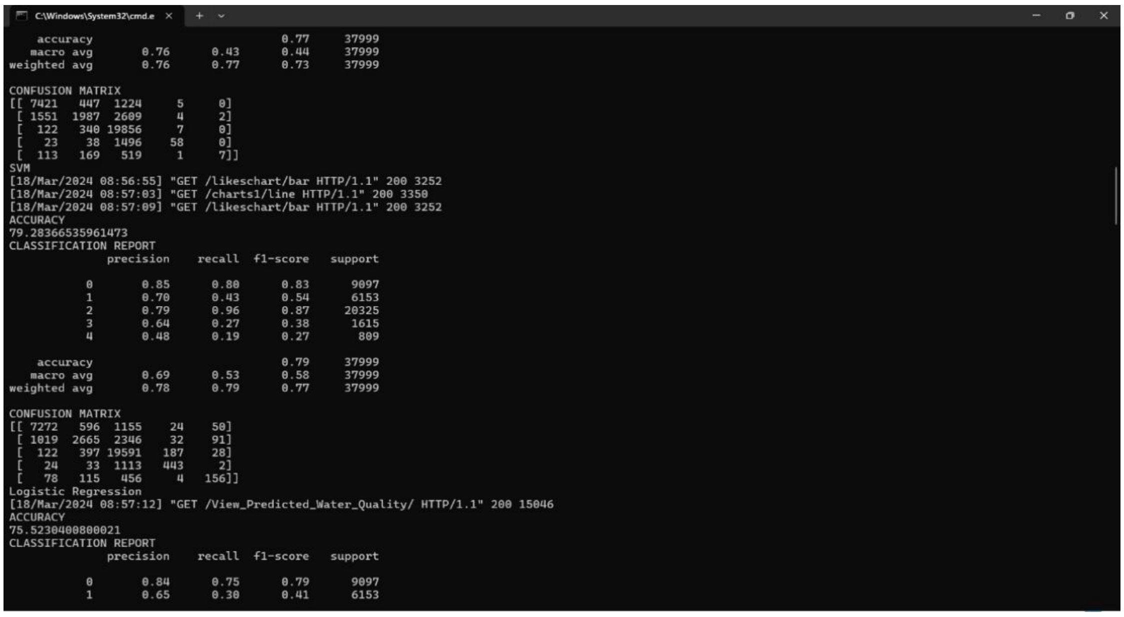

Service Provider
127.0.0.1:8000/charts/line
Predicting Urban Water Quality With Ubiquitous Data - A Data-Driven Approach
Train and Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Predicted Water Quality Type
Find Water Quality Prediction Ratio
Download Trained Data Sets
View Water Quality Prediction Ratio Results
View All Remote Users
Logout
PIE CHAERT
LINE CHART
Salinity 25.00%
Fluoride 12.50%
Iron 50.00%
Arsenic-Fully Polluted 12.50%
Salinity
Fluoride
Iron
Arsenic-Fully Polluted
10
15
20
25
30
35
40
45
50
55
CanvasJS.com

Figure 7.9 LINE CHART REPRESENTATION



```
    accuracy                           0.77     37999
   macro avg       0.76      0.43      0.44     37999
weighted avg       0.76      0.77      0.73     37999

CONFUSION MATRIX
[[ 7421   447  1224     5     0]
 [ 1551  1987  2609     4     2]
 [  122   340 19856     7     0]
 [   23    38  1496    58     0]
 [  113   169   519     1     7]]
SVM
[18/Mar/2024 08:56:55] "GET /likeschart/bar HTTP/1.1" 200 3252
[18/Mar/2024 08:57:03] "GET /charts1/line HTTP/1.1" 200 3350
[18/Mar/2024 08:57:09] "GET /likeschart/bar HTTP/1.1" 200 3252
ACCURACY
79.28366535961473
CLASSIFICATION REPORT
              precision    recall  f1-score   support

           0       0.85      0.80      0.83      9097
           1       0.70      0.43      0.54      6153
           2       0.79      0.96      0.87     20325
           3       0.64      0.27      0.38      1615
           4       0.48      0.19      0.27       809

    accuracy                           0.79     37999
   macro avg       0.69      0.53      0.58     37999
weighted avg       0.78      0.79      0.77     37999

CONFUSION MATRIX
[[ 7272   596  1155    24    50]
 [ 1019  2665  2346    32    91]
 [  122   397 19591   187    28]
 [   24    33  1113   443     2]
 [   78   115   456     4   156]]
Logistic Regression
[18/Mar/2024 08:57:12] "GET /View_Predicted_Water_Quality/ HTTP/1.1" 200 15046
ACCURACY
75.5230400800021
CLASSIFICATION REPORT
              precision    recall  f1-score   support

           0       0.84      0.75      0.79      9097
           1       0.65      0.30      0.41      6153
```

Figure 7.10 COMMAND LINE RESULTS

8. CONCLUSION

The project "Predicting Urban Water Quality With Ubiquitous Data – A Data-Driven Approach" represents a pioneering effort to revolutionize urban water management through the integration of advanced data analytics and ubiquitous data sources. By leveraging diverse data streams, ranging from environmental sensors to social media inputs, this project aims to provide real-time insights into water quality dynamics, thereby enabling proactive interventions and informed decision-making.

Through the development of predictive models and decision support systems, this project empowers urban planners, water authorities, and policymakers to anticipate and address water quality challenges effectively. By identifying pollution hotspots, discerning temporal trends, and analyzing the impact of anthropogenic activities, stakeholders can implement targeted strategies to mitigate pollution, safeguard public health, and preserve aquatic ecosystems.

Furthermore, this project underscores the transformative potential of data-driven approaches in fostering sustainable development and resilience in cities. By harnessing the wealth of data generated in urban environments, we can not only enhance water quality but also promote more efficient resource allocation, mitigate environmental risks, and enhance the overall quality of life for urban residents.

9. FUTURE ENHANCEMENT

In the continuous pursuit of refining urban water quality prediction through ubiquitous data, future advancements are poised to significantly enhance our understanding and management capabilities. A critical area for improvement lies in data fusion techniques, where sophisticated methodologies can be developed to seamlessly integrate the vast array of available data sources. By harmonizing data from water quality sensors, satellite imagery, social media, and municipal databases, these advancements aim to elevate the accuracy and reliability of predictive models, providing deeper insights into the complexities of urban water systems.

Emerging technologies offer another avenue for future enhancement, presenting opportunities to revolutionize the predictive modeling framework. Unmanned aerial vehicles (UAVs) equipped with specialized sensors hold promise for delivering high-resolution spatial data, particularly beneficial for monitoring water quality in remote or inaccessible areas. Furthermore, the integration of blockchain technology could fortify data security and transparency, instilling greater trust and confidence in the data sharing and validation processes essential for robust predictive modeling.

Real-time adaptive modeling stands out as a frontier ripe for exploration, offering the potential for dynamic adjustments to predictive models based on evolving environmental conditions. By continuously updating models in response to incoming data streams, these techniques could significantly enhance the timeliness and accuracy of water quality forecasts. This real-time responsiveness holds the key to more proactive management strategies, enabling swift interventions in the event of pollution incidents and bolstering resilience in urban water systems.

10.REFERENCES

[1] W. H. Organization, Guidelines for drinking-water quality, 2004, vol. 3.

[2] L. A. Rossman, R. M. Clark, and W. M. Grayman, “Modeling chlorine residuals in drinking-water distribution systems,” Journal of environmental engineering, vol. 120, no. 4, pp. 803–820, 1994.

[3] Y. Zheng, “Methodologies for cross-domain data fusion: An overview,” IEEE Transactions on Big Data, vol. 1, no. 1, pp. 16–34, 2015.

[4] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, “Urban computing: Concepts, methodologies, and applications,” ACM Transactions on Intelligent Systems and Technology, vol. 5, no. 3, pp. 38:1–38:55, 2014.

[5] Y. Zheng, H. Zhang, and Y. Yu, “Detecting collective anomalies from multiple spatio-temporal datasets across different domains,” 2015.

[6] Y. Liu, Y. Zheng, Y. Liang, S. Liu, and D. S. Rosenblum, “Urban water quality prediction based on multi-task multi-view learning,” in Proceedings of the International Joint Conference on Artificial Intelligence, 2016.

[7] H. Cohen, “Free chlorine testing,” http://www.cdc.gov/safewater/chlorineresidual - testing.html, 2014, accessed on 5 August 2016.

[8] Kim, D., Cho, S. K., Lee, J., & Kim, H. C. (2020). Urban water quality prediction using machine learning models: A review and perspectives. Water, 12(5), 1267.

[9] P. Castro and M. Neves, “Chlorine decay in water distribution systems case study–lousada network,” Electronic Journal of Environmental, Agricultural and Food Chemistry, vol. 2, no. 2, pp. 261–266, 2003.

[10] L. W. Mays, Water distribution system handbook, 1999. Knowledge Discovery, 2003, pp. 2–11.

[11] L. A. Rossman and P. F. Boulos, “Numerical methods for modelling water quality in distribution systems: A comparison,” Journal of Water Resources planning and management, vol. 122, no. 2, pp. 137–146, 1996.

[12] W. M. Grayman, R. M. Clark, and R. M. Males, “Modeling distributionsystem water quality: dynamic approach,” Journal of Water Resources Planning and Management, vol. 114, no. 3, pp. 295–312, 1988.

[13] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, “A symbolic representation of time series, with implications for streaming algorithms,” in Proceedings of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003, pp. 2–11.

[14] G. Luo, K. Yi, S.-W. Cheng, Z. Li, W. Fan, C. He, and Y. Mu, “Piecewise linear approximation of streaming time series data with max-error guarantees,” in Proceedings of the IEEE International Conference on Data Engineering, 2015, pp. 173–184.

[15] Zhang, K., Liu, X., Li, X., & Wu, S. (2021). A review of data-driven models for predicting urban water quality: Challenges, opportunities, and future directions. Environmental Research, 197, 111073.