K.RAMAKRISHNAN COLLEGE OF ENGINEERING
An Autonomous Institution
Samayapuram, Tiruchirappalli – 621 112, Tamilnadu, India.

A Project Report

on

WATER MANAGEMENT SYSTEM

Submitted in partial fulfillment of requirements for the award of the course

of

MGB1201 – PYTHON PROGRAMMING

Under the guidance of

Mrs.S.RAJESWARI M.E.

Assistant Professor / CSE

Submitted By

KARTHIKEYAN K (8115U23ME022)

DEPARTMENT OF MECHANICAL ENGINEERING

K.RAMAKRISHNAN COLLEGE OF ENGINEERING
(Autonomous)

TRICHY-621 112

DECEMBER 2024

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

(Autonomous Institution affiliated to Anna University, Chennai)

TRICHY-621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"WATER MANAGEMENT SYSTEM"** is the bonafide work of **KARTHIKEYAN K (8115U23ME022)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE
Dr. T. M. NITHYA, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
ASSOCIATE PROFESSOR
Department of CSE
K.Ramakrishnan College of Engineering (Autonomous)
Samayapuram–621112.

SIGNATURE
Mrs.S.RAJESWARI M.E.
SUPERVISOR
ASSISTANT PROFESSOR
Department of CSE
K.Ramakrishnan College of Engineering (Autonomous)
Samayapuram–621112.

Submitted for the End Semester Examination held on…….

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

I declare that the project report on **"WATER MANAGEMENT SYSTEM"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **MGB1201 – PYTHON PROGRAMMING**

.

Signature

_____

KARTHIKEYAN K

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Engineering (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. D. SRINIVASAN, B.E, M.E., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. T. M. NITHYA, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs.S.RAJESWARI M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE INSTITUTION

To achieve a prominent position among the top technical institutions

MISSION OF THE INSTITUTION

M1: To bestow standard technical education par excellence through state of the art infrastructure, competent faculty and high ethical standards.

M2: To nurture research and entrepreneurial skills among students in cutting edge technologies.

M3: To provide education for developing high-quality professionals to transform the society.

VISION OF THE DEPARTMENT

To create eminent professionals of Computer Science and Engineering by imparting quality education.

MISSION OF THE DEPARTMENT

M1: To provide technical exposure in the field of Computer Science and Engineering through state of the art infrastructure and ethical standards.

M2: To engage the students in research and development activities in the field of Computer Science and Engineering.

M3: To empower the learners to involve in industrial and multi-disciplinary projects for addressing the societal needs.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

Our graduates shall

PEO1: Analyse, design and create innovative products for addressing social needs.

PEO2: Equip themselves for employability, higher studies and research.

PEO3: Nurture the leadership qualities and entrepreneurial skills for their successful career.

PROGRAM OUTCOMES

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write

11. effective reports and design documentation, make effective presentations, and give and receive clear instructions.
12. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
13. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1:** Apply the basic and advanced knowledge in developing software, hardware and firmware solutions addressing real life problems.

- **PSO2:** Design, develop, test and implement product-based solutions for their career enhancement.

ABSTRACT

This Python script is designed as a foundational component for a water management system. It includes configuration parameters for low and high water level thresholds and sets up logging for informative messages with timestamps.

The script maintains a global variable to track the state of the pump (either "ON" or "OFF") and provides utility functions for logging messages and controlling the pump. The main logic of the script revolves around the water_ management_ system() function, which periodically reads the water level from a simulated sensor. Based on the current water level, it determines whether to turn the pump on, off, or leave it unchanged within predefined thresholds.

Relevant actions and state changes are logged accordingly. Exception handling is implemented to gracefully handle user interrupts, ensuring proper cleanup of resources, such as GPIO settings. Upon execution, the water_ management_ system() function is invoked to initiate the monitoring and control process.

ABSTRACT WITH POs AND PSOs MAPPING

ABSTRACT	POs MAPPED	PSOs MAPPED
This Python script is designed as a foundational component for a water management system. It includes configuration parameters for low and high water level thresholds and sets up logging for informative messages with timestamps. The script maintains a global variable to track the state of the pump (either "ON" or "OFF") and provides utility functions for logging messages and controlling the pump. The main logic of the script revolves around the water_ management_ system() function, which periodically reads the water level from a simulated sensor. Based on the current water level, it determines whether to turn the pump on, off, or leave it unchanged within predefined thresholds. Relevant actions and state changes are logged accordingly. Exception handling is implemented to gracefully handle user interrupts, ensuring proper cleanup of resources, such as GPIO settings. Upon execution, the water_ management_ system() function is invoked to initiate the monitoring and control process.	**PO1-1** **PO2-2** **PO3-1** **PO12-2**	**PSO1-2**

Note: 1- Low, 2-Medium, 3- High

SUPERVISOR **HEAD OF THE DEPARTMENT**

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Objective

Efficient water management is a pressing need in modern society, and this Python-based project aims to develop a system that promotes the judicious use of water resources. The primary objective is to create a user-friendly platform that tracks and monitors water consumption across various sectors, such as residential, agricultural, and industrial. By utilizing a database to store and retrieve data, the system ensures accurate logging of water usage, enabling users to analyze consumption patterns and implement measures to reduce wastage effectively.

This project also focuses on providing features for alerting users about abnormal water usage, such as leaks or overconsumption. By integrating threshold limits and notification systems, the project enables users to address issues promptly, thereby reducing unnecessary water loss. Furthermore, the system is designed to offer detailed reports, charts, and suggestions based on consumption trends, helping stakeholders make informed decisions about water conservation strategies.

In addition to tracking and alerts, the system encourages sustainable practices by facilitating the efficient allocation of water resources. For instance, it can prioritize water distribution during shortages or monitor reservoir levels to ensure equitable supply. With scalability and adaptability in mind, the project provides a foundation for integrating advanced features such as IoT-based sensors for real-time monitoring or predictive analytics to forecast future water needs, ensuring the system remains relevant for long-term water management efforts.

Overview

The Water Management System in Python is a comprehensive solution designed to address the critical issue of water resource management in various sectors, including residential, agricultural, and industrial. The system leverages Python's robust programming capabilities to monitor, track, and analyze water consumption efficiently. With a focus on user-friendliness and functionality, it incorporates a database-driven approach to store and retrieve water usage data, allowing users to identify patterns and adopt effective water-saving measures.

One of the key features of this system is its ability to alert users about anomalies such as excessive usage or potential leaks. By defining usage thresholds and implementing notification systems, the project aims to minimize water wastage and improve resource efficiency. Additionally, the system generates detailed reports and visual representations of water usage trends, equipping stakeholders with the information needed to make informed decisions about water allocation and conservation.

This project is designed to be scalable and future-ready, providing a framework that can integrate advanced technologies such as IoT sensors for real-time monitoring or machine learning algorithms for predictive analytics. By promoting sustainable water usage practices and improving resource management, the system addresses current challenges while laying the groundwork for long-term solutions in water conservation.

Python Programming Concepts

The Water Management System leverages Python's list and string concepts to efficiently manage and present data. Lists are used to store and organize water usage records for various sectors or time intervals, enabling easy data manipulation. By iterating through lists, the system calculates metrics such as total and average consumption or identifies anomalies like excessive usage. This ensures that users can analyze their water usage patterns effectively and make informed decisions. Lists also provide a scalable solution for handling large datasets, accommodating the addition or removal of records dynamically as needed.

Strings play a vital role in generating user-friendly alerts, summaries, and reports. For instance, when water usage exceeds a predefined limit, the system uses string formatting to create personalized warning messages. Strings are also combined with list data to produce comprehensive reports that detail daily or weekly consumption. This integration of strings ensures that the information is not only functional but also easy for users to interpret, enhancing the system's usability. Together, lists and strings form the backbone of the system's data processing and communication functionalities.

CHAPTER 2
PROJECT METHODOLOGY

Proposed Work

The proposed work for the Water Management System focuses on developing an integrated platform that enables efficient monitoring, management, and analysis of water consumption across different sectors, such as residential, commercial, and industrial. The system will allow users to input, store, and analyse their water usage data, helping them track consumption trends over time. By utilizing a simple database structure, the system will allow easy updates, additions, and retrieval of water usage records, enabling users to maintain accurate records for water conservation and billing purposes.

Using Python programming concepts, the system will process and analyse data stored in lists, providing users with real-time insights into their consumption patterns. The water usage data can be displayed as reports, summaries, and alerts through string formatting, offering an intuitive interface for users. Additionally, users will be notified when their water consumption exceeds predefined limits, prompting them to take corrective actions to prevent wastage. This real-time feedback mechanism is designed to help users become more conscious of their water usage habits.

In the future, the system could be enhanced by integrating IoT sensors for real-time data collection, enabling automated tracking of water flow and usage. Machine learning algorithms can also be introduced to predict future water consumption based on historical data, providing users with projections and recommendations. The system will be continuously updated and refined to improve functionality, ensuring that it supports sustainable water management practices and contributes to long-term water conservation efforts.

Block Diagram

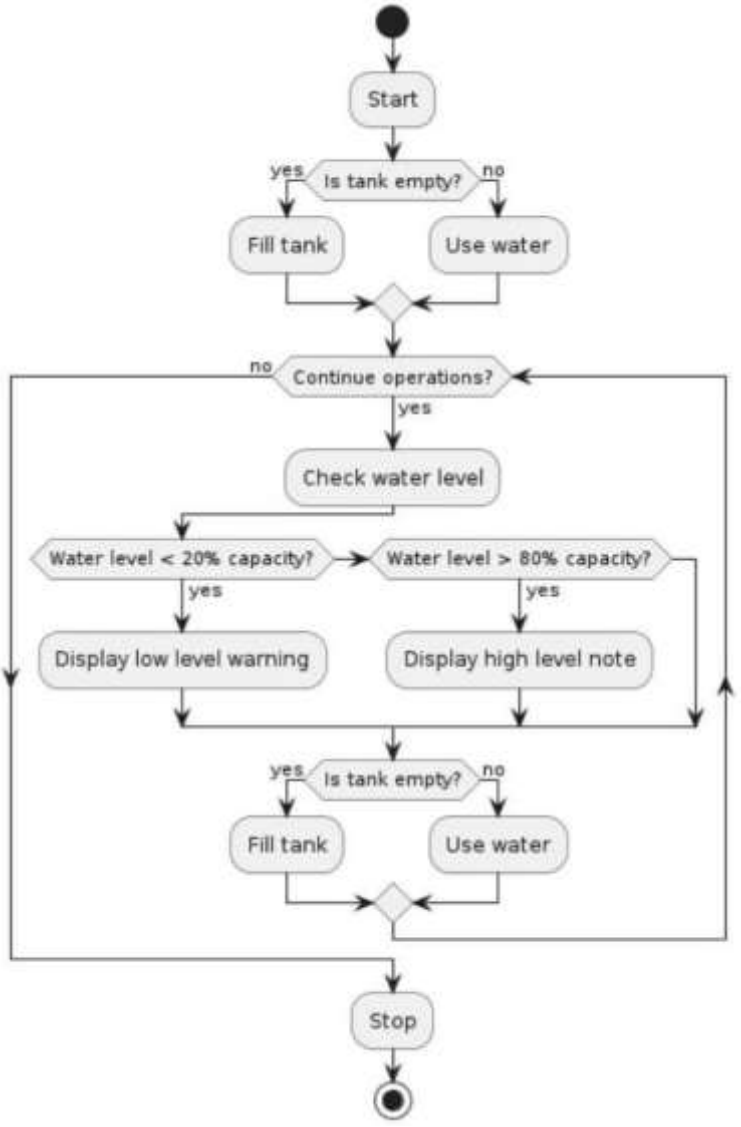


Fig 2.1: flow diagram

CHAPTER -3
MODULUS

MODULE 3.1: cleanup_gpio()

The cleanup_gpio() function is commonly used in Python programs that interact with GPIO (General Purpose Input/Output) pins on devices like Raspberry Pi. It's a method from the RPi.GPIO libraryand is used to reset the state of the GPIO pins.

MODULE 3..2: read_water_level()

The read_water_level() function in Python would typically interact with a water level sensor tomeasure and return the water level. Its implementation depends on the type of sensor being used, such

as a **digital water level sensor** (high/low) or an **ultrasonic sensor** (precise distance measurement).

MODULE3.3: pump_control():

The pump_control() function in a water management system typically manages the operation of a

water pump based on the water level or other system parameters. The function can control thepumctivation or deactivation using a relay module connected to a GPIO pin.

MODULE3.4: current_time():

In Python, you can get the current time using the datetime module. Here's how you

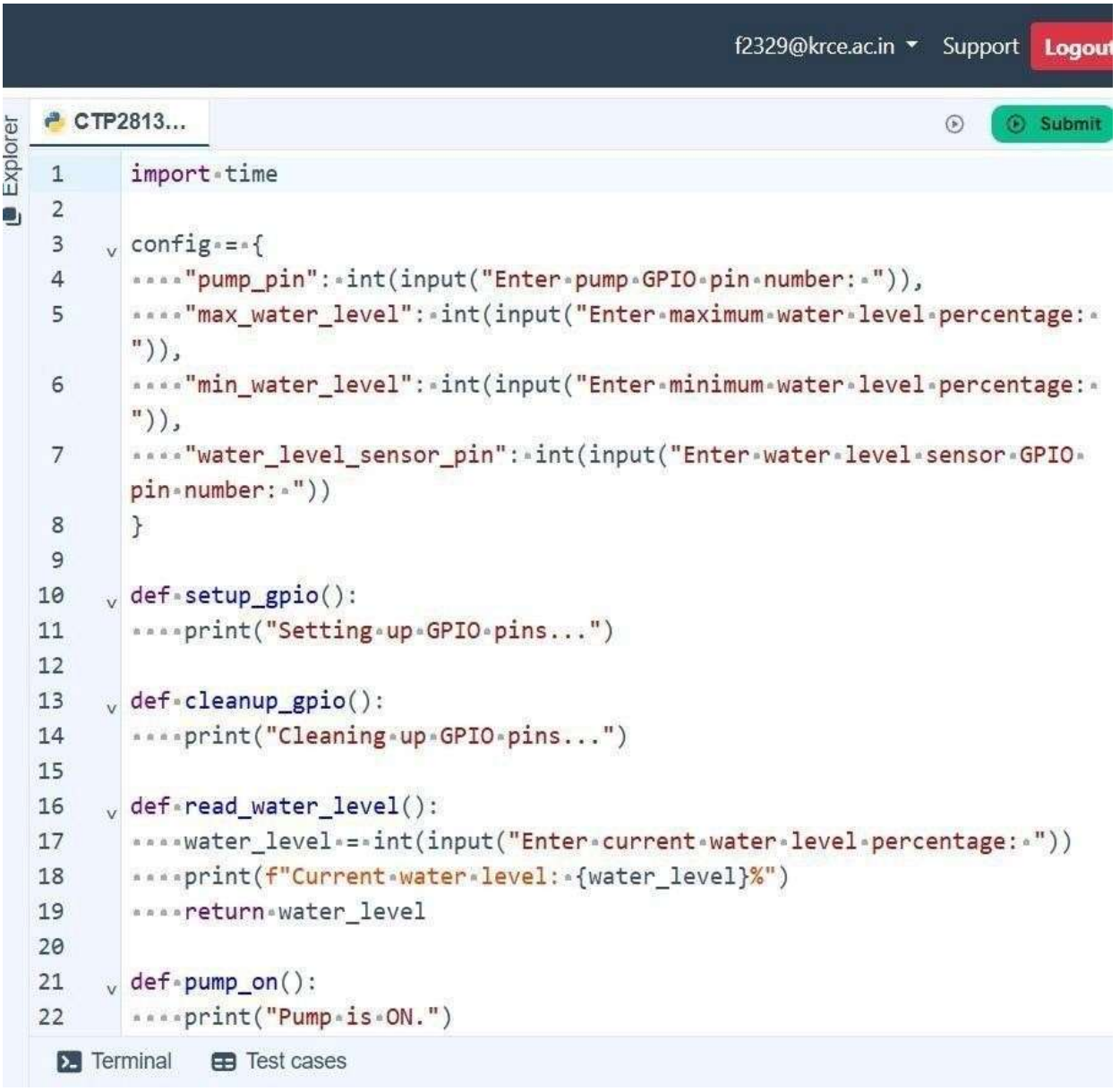
can implement acurrent_time() function

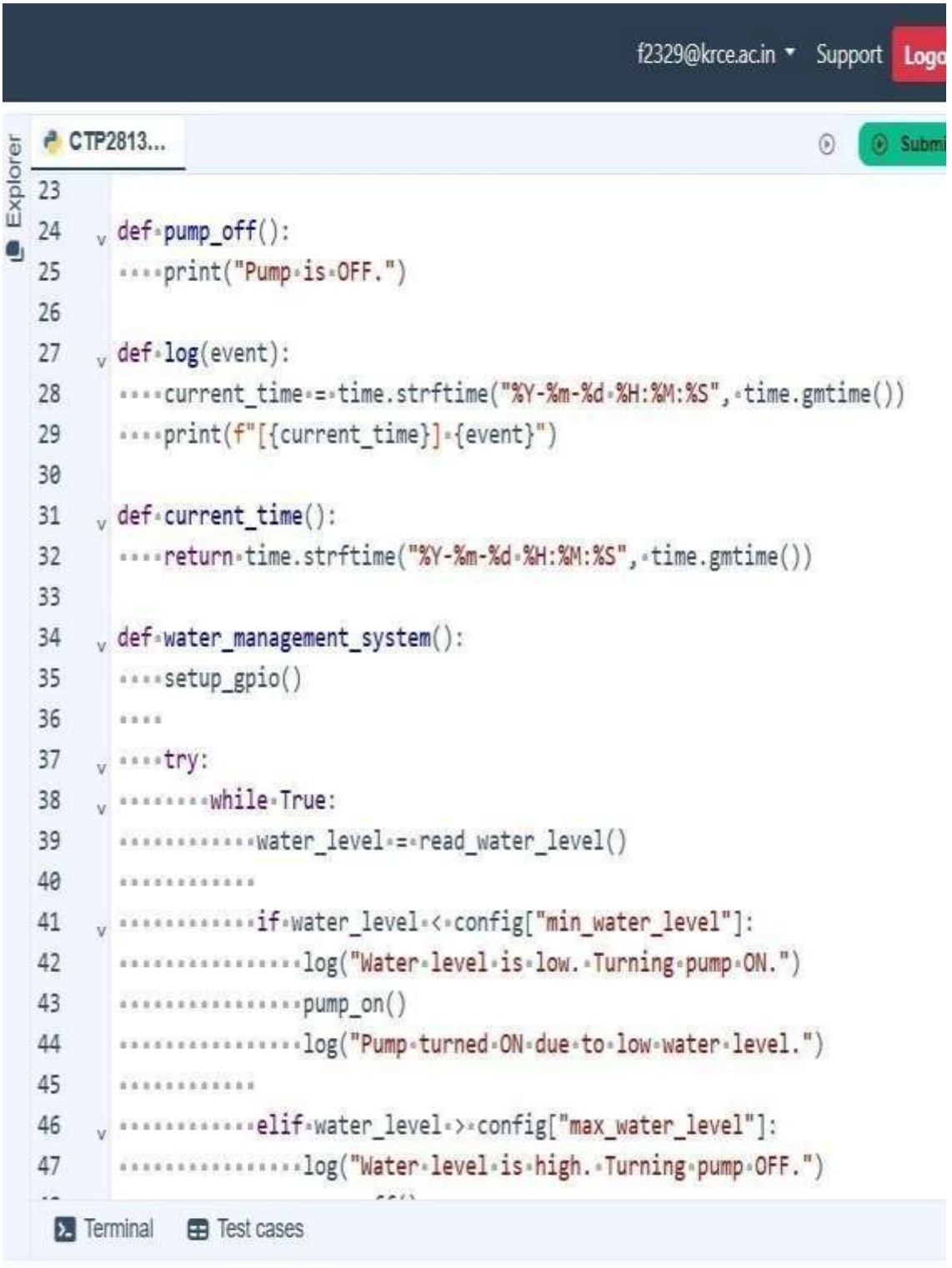
MODULE3. 5 : setup_gpio():

The setup_gpio() function is used to initialize GPIO pins for input or output in Python when workingwith hardware like a Raspberry Pi. It simplifies the GPIO setup process, ensuring proper configuration for sensors, relays, or other connected components.

CHAPTER 4

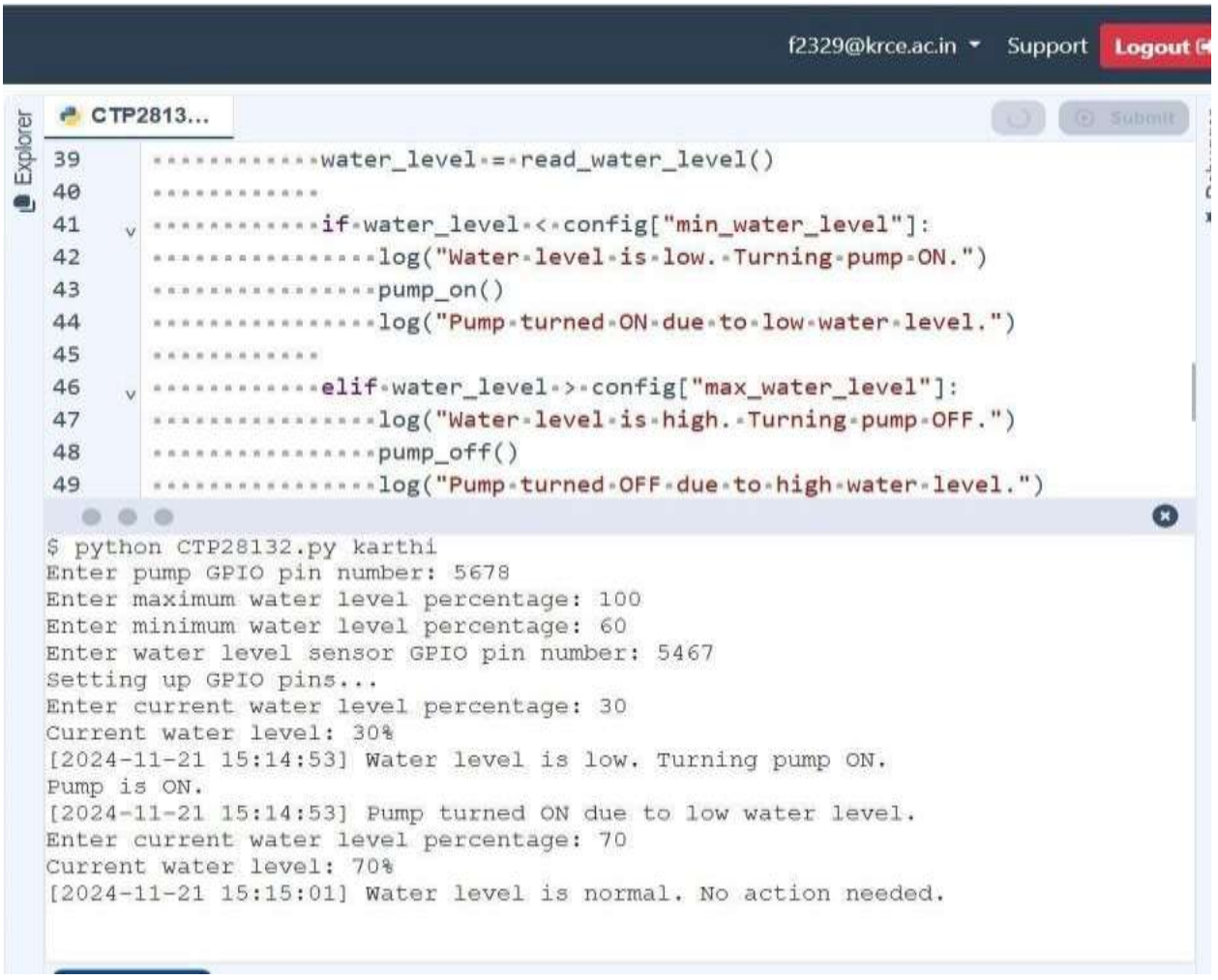
RESULTS AND DISCUSSION






```
23
24  def pump_off():
25      print("Pump is OFF.")
26
27  def log(event):
28      current_time = time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())
29      print(f"[{current_time}] {event}")
30
31  def current_time():
32      return time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())
33
34  def water_management_system():
35      setup_gpio()
36
37      try:
38          while True:
39              water_level = read_water_level()
40
41              if water_level < config["min_water_level"]:
42                  log("Water level is low. Turning pump ON.")
43                  pump_on()
44                  log("Pump turned ON due to low water level.")
45
46              elif water_level > config["max_water_level"]:
47                  log("Water level is high. Turning pump OFF.")
```

Result:


f2329@krce.ac.in
Support
Logout
Explorer
CTP2813...
Submit
39 ····water_level·=·read_water_level()
40 ····
41 ····if·water_level·<·config["min_water_level"]:
42 ····log("Water·level·is·low.·Turning·pump·ON.")
43 ····pump_on()
44 ····log("Pump·turned·ON·due·to·low·water·level.")
45 ····
46 ····elif·water_level·>·config["max_water_level"]:
47 ····log("Water·level·is·high.·Turning·pump·OFF.")
48 ····pump_off()
49 ····log("Pump·turned·OFF·due·to·high·water·level.")
$ python CTP28132.py karthi
Enter pump GPIO pin number: 5678
Enter maximum water level percentage: 100
Enter minimum water level percentage: 60
Enter water level sensor GPIO pin number: 5467
Setting up GPIO pins...
Enter current water level percentage: 30
Current water level: 30%
[2024-11-21 15:14:53] Water level is low. Turning pump ON.
Pump is ON.
[2024-11-21 15:14:53] Pump turned ON due to low water level.
Enter current water level percentage: 70
Current water level: 70%
[2024-11-21 15:15:01] Water level is normal. No action needed.

CHAPTER 5
CONCLUSION

The Configuration Model module plays a crucial role in the flexibility and adaptability of the Water Management System. It enables the system to store and retrieve configuration settings, such as water usage limits, alert thresholds, and database file paths, without the need for hardcoded values. By storing these configurations in a JSON file, it ensures that the system's settings are easily manageable and can be modified or updated without altering the core application logic. This approach improves the maintainability and scalability of the system, allowing for a dynamic configuration setup.

Furthermore, the module simplifies the process of managing system parameters by offering easy-to-use methods for accessing and updating configuration data. Users can customize essential settings, such as water usage limits, according to their specific needs, and these changes are automatically saved in the configuration file. This reduces the risk of errors and ensures that all configurations are consistent across different sessions of the application. It also allows the system to adapt quickly to evolving requirements, especially in dynamic environments where system parameters may change frequently.

In conclusion, the Configuration Model enhances the Water Management System's overall functionality by providing a centralized and organized way to manage application settings. It streamlines the process of customizing and maintaining the system, ensuring that the system can efficiently handle updates, changes, and new feature integrations. By separating configuration from the core logic, the module supports better modularity, offering long-term benefits for system scalability and ease of maintenance.

REFERENCES:

"Fluent Python" by Luciano Ramalho

This book provides in-depth insights into Python programming, including advanced techniques for writing clean, maintainable code. While not specifically about configuration management, it helps developers understand best practices for organizing Python projects, which could be useful for implementing configuration models.

"Python 3 Object-Oriented Programming" by Dusty Phillips

This book covers object-oriented programming in Python, including how to structure and organize code effectively. It could help in understanding how to design modules like the Configuration Model for scalability and maintainability.

"Python Cookbook" by David Beazley and Brian K. Jones

A practical guide for Python developers, this book provides recipes for common tasks, including configuration management and working with data formats like JSON. It can be helpful for implementing configuration files in Python.

"Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides

Although it focuses on design patterns for object-oriented software, this book provides valuable insights into modularity, scalability, and flexibility in software design, which are key principles behind the Configuration Model.

"Learning Python" by Mark Lutz

This is a comprehensive guide to learning Python. It covers a wide range of topics, including working with files, modules, and dictionaries, all of which are useful when building a configuration model that reads and writes settings from external files like JSON.

APPENDIX

(Coding)

```
import time
def cleanup_gpio():
print("Cleaning up GPIO pins...")


def read_water_level():
water_level = int(input("Enter current water level percentage: "))
print(f"Current water level: {water_level}%")
return water_level


def pump_on():
print("Pump is ON.")


def pump_off():
print("Pump is OFF.")


def log(event):
current_time = time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())
print(f"[{current_time}] {event}")


def current_time():
return time.strftime("%Y-%m-%d %H:%M:%S", time.gmtime())


def water_management_system():
```

```
setup_gpio()

try:

while True:

water_level = read_water_level()
if water_level < config["min_water_level"]:

log("Water level is low. Turning pump ON.")

pump_on()

log("Pump turned ON due to low water level.")

elif water_level > config["max_water_level"]:

log("Water level is high. Turning pump OFF.")

pump_off()

log("Pump turned OFF due to high water level.")

else:

log("Water level is normal. No action needed.")

time.sleep(5)

except KeyboardInterrupt:
log("Water management system interrupted.")

finally:

cleanup_gpio()

log("Water management system shutdown.")


water_management_system()
```