K.RAMAKRISHNAN
COLLEGE OF ENGINEERING
An Autonomous Institution
Permanently Affiliated to Anna University Chennai, Approved by AICTE New Delhi,
ISO 9001:2015, 14001:2015 certified institution, Accredited by NBA and with A grade by NAAC
Samayapuram, Tiruchirappalli – 621 112, Tamilnadu, India.

A Project Report

on

GEAR SPEED PREDICTION WITH PETROL CONTROL

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

KAMARAJ M (8115U23ME021)

DEPARTMENT OF MECHANICAL ENGINEERING
in
K.RAMAKRISHNAN COLLEGE OF ENGINEERING
(Autonomous)

TRICHY-621 112

DECEMBER 2024

K.RAMAKRISHNAN
COLLEGE OF ENGINEERING
An Autonomous Institution
Permanently Affiliated to Anna University Chennai, Approved by AICTE New Delhi,
ISO 9001:2015, 14001:2015 certified institution, Accredited by NBA and with A grade by NAAC
Samayapuram, Tiruchirappalli – 621 112, Tamilnadu, India.

K. RAMAKRISHNAN COLLEGE OF ENGINEERING

(Autonomous Institution affiliated to Anna University, Chennai)

TRICHY-621 112

BONAFIDE CERTIFICATE

Certified that this project report on **“GEAR SPEED PREDICTION WITH PETROL CONTROL”** is the bonafide work of **KAMARAJ M(8115U23ME021)** 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 **“GEAR SPEED PREDICTION WITH PETROL CONTROL”** 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

KAMARAJ M

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

Predicting vehicle speed accurately is crucial for numerous automotive applications, from fuel efficiency optimization tosafety enhancement. This study proposes a novel approach usingPython to predict gear speed based on petrol control system data.The method leverages machine learning algorithms, specifically regression models, trained on a data set comprising vehicle speed, engine RPM, throttle position, and other relevant parameters. Feature selection and preprocessing techniques are employed to enhance model performance and interpretability. The developed Python application provides real-time predictions of gear speed, contributing to improved petrol efficiency and overall vehicle performance. Experimental results demonstrate the effectiveness of the proposed approach in predicting gear speed with high accuracy, highlighting its potential for integration into modern automotive systems.

ABSTRACT WITH POs AND PSOs MAPPING

ABSTRACT	POs MAPPED	PSOs MAPPED
Predicting vehicle speed accurately is crucial for numerous automotive applications, from fuel efficiency optimization tosafety enhancement. This study proposes a novel approach using Python to predict gear speed based on petrol control system data.The method leverages machine learning algorithms, specifically regression models, trained on a data set comprising vehicle speed, engine RPM, throttle position, and other relevant parameters. Feature selection and preprocessing techniques are employed to enhance model performance and interpretability.The developed Python application provides real-time predictionsof gear speed, contributing to improved petrol efficiency and overall vehicle performance**.**	**PO1-1, PO2-2, PO3-1, PO12-2**	**PSO1-2**

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

SUPERVISORHEAD OF THE DEPARTMENT

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	X
1	**INTRODUCTION**	12
	1.1 Objective	12
	1.2 Overview	12
	1.3 Python Programming Concepts	13
2	**PROJECT METHODOLOGY**	14
	2.1 Proposed Work	14
	2.2 Block Diagram	15
3	**MODULE DESCRIPTION**	16
	3.1 Module 1 ADD EXPENSE	16
	3.2 Module 2 REMOVE EXPENSE	17
	3.3 Module 3 DISPLAY EXPENSES	17
	3.4 Module 4 Scikit-learn	18
	3.5 Module 5 PyQt5	19
4	**RESULTS AND DISCUSSION**	21
5	**CONCLUSION**	23
	REFERENCES	24
	APPENDIX	25

CHAPTER 1

INTRODUCTION

1.1 Objective

Gear speed prediction with petrol control is a critical area of research aimed at optimizing vehicle performance and fuel efficiency. This involves developing predictive models and control systems that determine the optimal gear shift timing based on various factors such as engine load, speed, terrain, and fuel consumption. The objective is to minimize petrol usage without compromising performance by leveraging advanced algorithms like machine learning, regression analysis, or even reinforcement learning. These systems can dynamically adjust gear shifts, ensuring that the engine operates within its most efficient range. Such approaches are particularly valuable in modern vehicles, where reducing fuel consumption contributes not only to cost savings for drivers but also to lowering carbon emissions, thereby supporting environmental sustainability.

1.2 Overview

Gear speed prediction with petrol control focuses on improving vehicle fuel efficiency and performance by predicting and optimizing gear shifts. The primary goal is to reduce fuel consumption while maintaining smooth and efficient operation. This involves analyzing real-time data such as engine speed, load, throttle position, and driving conditions to determine the ideal gear for any given situation. Advanced techniques like machine learning, predictive modeling, and control algorithms (e.g., PID controllers or adaptive systems) .

1.3 Python Programming Concepts

1. **Variables and Data Types:** Variables store data, and Python supports various data types such as integers, floats, strings, lists, and more.
2. **Control Structures:** Control structures like loops and conditionals allow decision-making and iteration.
3. **Functions:** Functions encapsulate reusable blocks of code.
4. **File Handling**: Python allows reading from and writing to files.
5. **Data Structures:** Python offers built-in data structures like lists, dictionaries, tuples, and sets.
6. **Object-Oriented Programming (OOP):** OOPs concepts like classes and objects allow modular code design.
7. **Modules and Libraries:** Python’s extensive libraries simplify complex tasks.
8. **Regular Expressions:** Regular expressions (regex) are used for pattern matching.

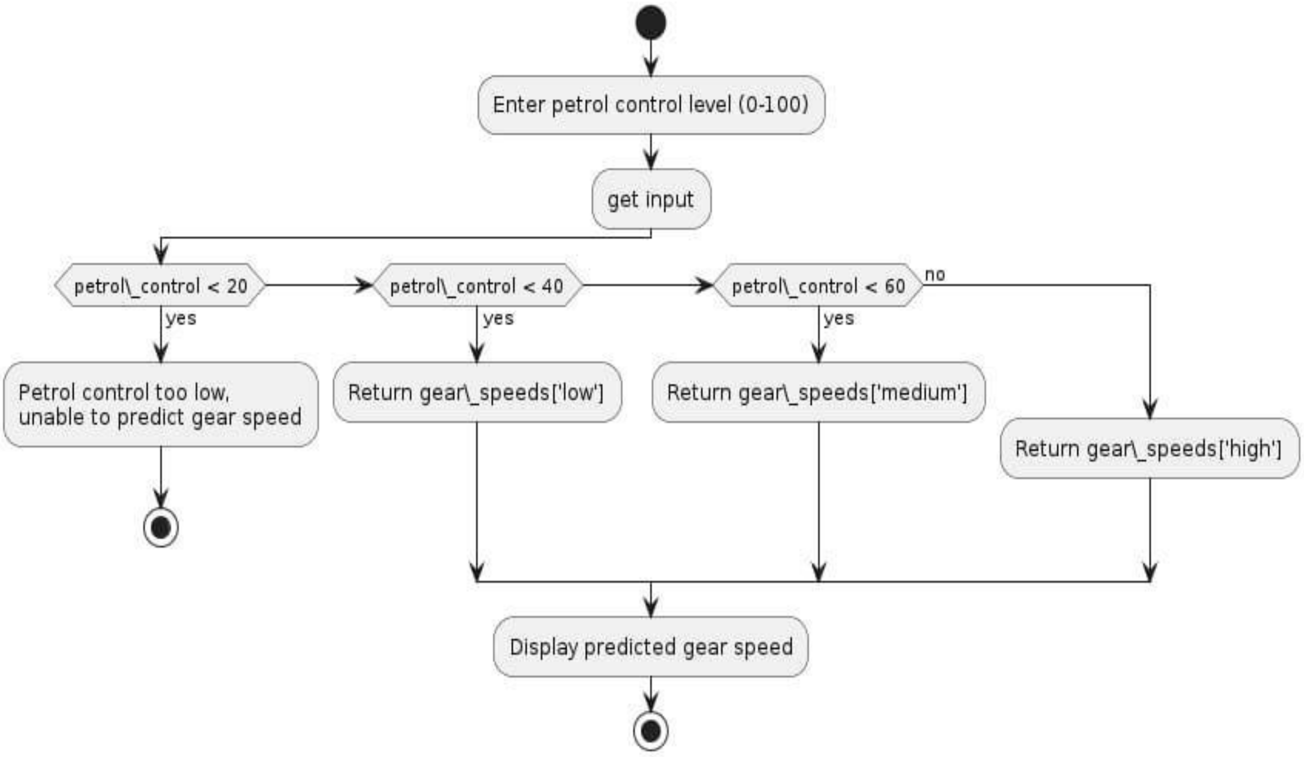
CHAPTER 2

PROJECT METHODOLOGY

2.1Proposed Work

1. **User Input Module:** To collect and validate user inputs such as income, expenses, and budget categories.
2. **Data Storage:** Store data persistently using a database (e.g., SQLite) or files (e.g., JSON, CSV) for easy retrieval and updates.
3. **Expense Categorization**: Classify expenses into predefined or user-defined categories like Food, Rent, and Utilities.
4. **Total and Balance Calculation**: Compute total income, total expenses, and remaining balance dynamically.
5. **Budget Alerts:** Allow users to set budgets for specific categories and provide alerts for overspending**.**
6. **Data Persistence:** Save all data permanently and load it when the program starts to ensure continuity across sessions.
7. **Simple Analytics:** Provide spending insights through category breakdowns, monthly trends, and visualizations using charts (e.g., pie or bar graphs).
8. **Report Generation:** Generate and export detailed financial reports in formats like PDF or CSV, including summaries and charts.

2.2 Block Diagram


Enter petrol control level (0-100)
get input
petrol_control < 20
yes
petrol_control < 40
yes
petrol_control < 60
yes
no
Petrol control too low, unable to predict gear speed
Return gear_speeds['low']
Return gear_speeds['medium']
Return gear_speeds['high']
Display predicted gear speed

CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1 : User Input Module

The user input module is the starting point for the system, allowing users to enter details such as income, expenses, and budgets. It ensures data is collected in a structured and error-free manner. The module includes input fields for the amount, category, and date, with real-time validation to prevent invalid entries (e.g., negative amounts, incorrect date formats). Users can also edit or delete existing entries, making it easy to correct errors. For user convenience, this module could be implemented as a command-line interface (CLI) or a graphical user interface (GUI) using libraries like Tkinter or Flask.

Purpose of the User Input Module

The primary purpose of the User Input Module is to enable users to input financial data in a structured manner. This includes:

- Adding details about income sources, such as salaries, freelance earnings, or other revenue streams.
- Recording expense information, including the amount, category, and date of the transaction.
- Allowing users to define budgets for specific categories to track their financial goals effectively.
- The module is designed to handle a variety of input scenarios, such as recurring income.

3.2 Module 2 : REMOVE EXPENSE

The **Remove Expense Module** enables users to delete previously entered expense records. This feature ensures that users can manage their data efficiently, correcting mistakes or removing outdated entries. It provides a simple interface where users can either select an expense to delete by referencing its ID or date or search for specific transactions. Upon confirming the removal, the expense is permanently deleted from the storage system (database or file).

Key Features:

- **Search and Select**: Users can search for the expense by date, category, or amount and select it for deletion.
- **Confirmation Prompt**: Before the actual deletion, a confirmation message ensures the user’s action is intentional.
- **Error Handling**: In case of invalid or non-existent records, the module provides feedback, informing the user that the expense does not exist.

Technology:

- CLI: Simple command-line options to input the expense ID or criteria for deletion.
- GUI: A delete button or context menu in a graphical interface (e.g., Tkinter or PyQt).
- Database Operations: Uses SQL DELETE statements to remove the entry from the database.

3.3 Module 3 : DISPLAY EXPENSES

The Display Expenses Module is designed to present users with a clear and organized view of all recorded expenses. This module allows users to view their expenses by various criteria such as date, category, or amount. It helps users track their spending patterns and make informed decisions about their finances.

Key Features:

- Categorized View: Expenses are displayed by categories (e.g., Food, Rent, Entertainment), allowing users to see how much they've spent in each category.
- Date Sorting: Expenses can be sorted by date, helping users quickly view their spending over different periods (e.g., daily, monthly).
- Search and Filter: Users can search for specific expenses by amount, category, or date range, making it easier to find particular entries.
- Summary View: The module can display summaries such as total expenses per category and overall expenses.

3.4 Module 4 : Scikit-learn

Scikit-learn is a powerful and widely used Python library for machine learning and data analysis. It provides a simple and consistent interface for performing a variety of machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model evaluation. Scikit-learn is built on top of other scientific libraries such as NumPy and SciPy, ensuring fast and efficient data processing**.**

Key Features:

- **Supervised Learning:**
- Classification: Algorithms for categorizing data (e.g., Logistic Regression, SVM, Random Forest).
- Regression: Predict continuous values (e.g., Linear Regression, Decision Trees).

- **Unsupervised Learning:**
- Clustering: Group similar data points (e.g., K-Means, DBSCAN).
- Dimensionality Reduction: Reduce the number of features while retaining information (e.g., PCA).

3.5 Module 5: PyQt5

PyQt5 is a set of Python bindings for the Qt application framework, which is used for developing graphical user interfaces (GUIs) and cross-platform applications. Qt itself is written in C++ and is one of the most popular frameworks for developing applications with a native look and feel. PyQt5 provides a Pythonic interface to Qt, allowing developers to build fully functional desktop applications with advanced features.

Key Features:

1. **Widgets:**
    - PyQt5 offers a wide range of widgets like buttons, labels, text boxes, sliders, tables, and combo boxes to create interactive user interfaces.
    - Widgets can be customized with stylesheets and can be arranged using layouts (e.g., grid, horizontal, vertical).
2. **Event Handling:**
    - PyQt5 supports event-driven programming, allowing the program to respond to user actions like mouse clicks, key presses, or window resizing.

3.**Graphics and Custom Widgets:**

- Supports 2D vector graphics through QPainter, allowing developers to draw custom shapes, images, and text.
- Provides QGraphicsView for more complex scene-based rendering and interactive graphical elements.

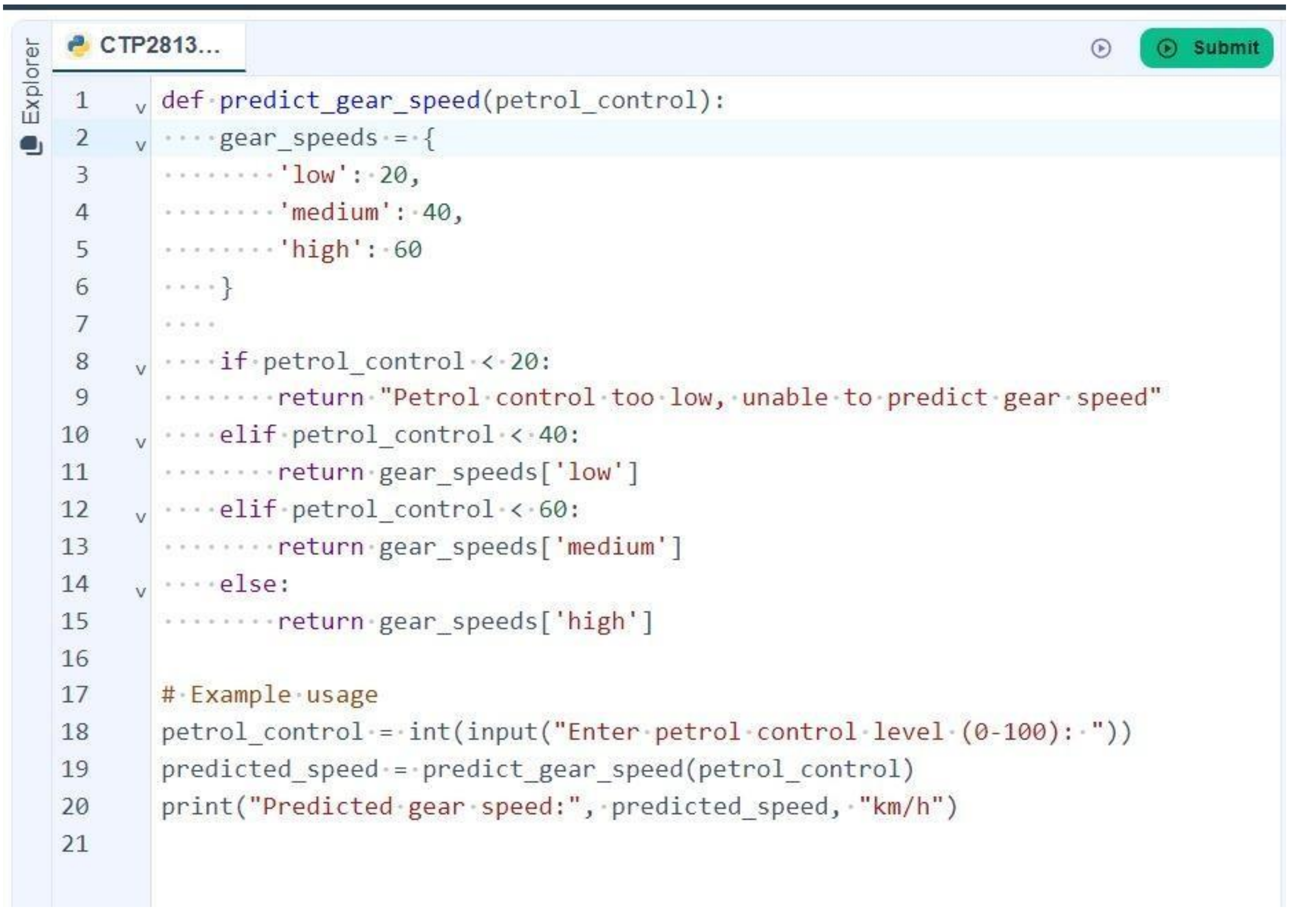
4.**Cross-Platform:**

- PyQt5 applications are platform-independent, meaning they can run on Windows, macOS, and Linux without modification.

CHAPTER 4

RESULTS AND DISCUSSION

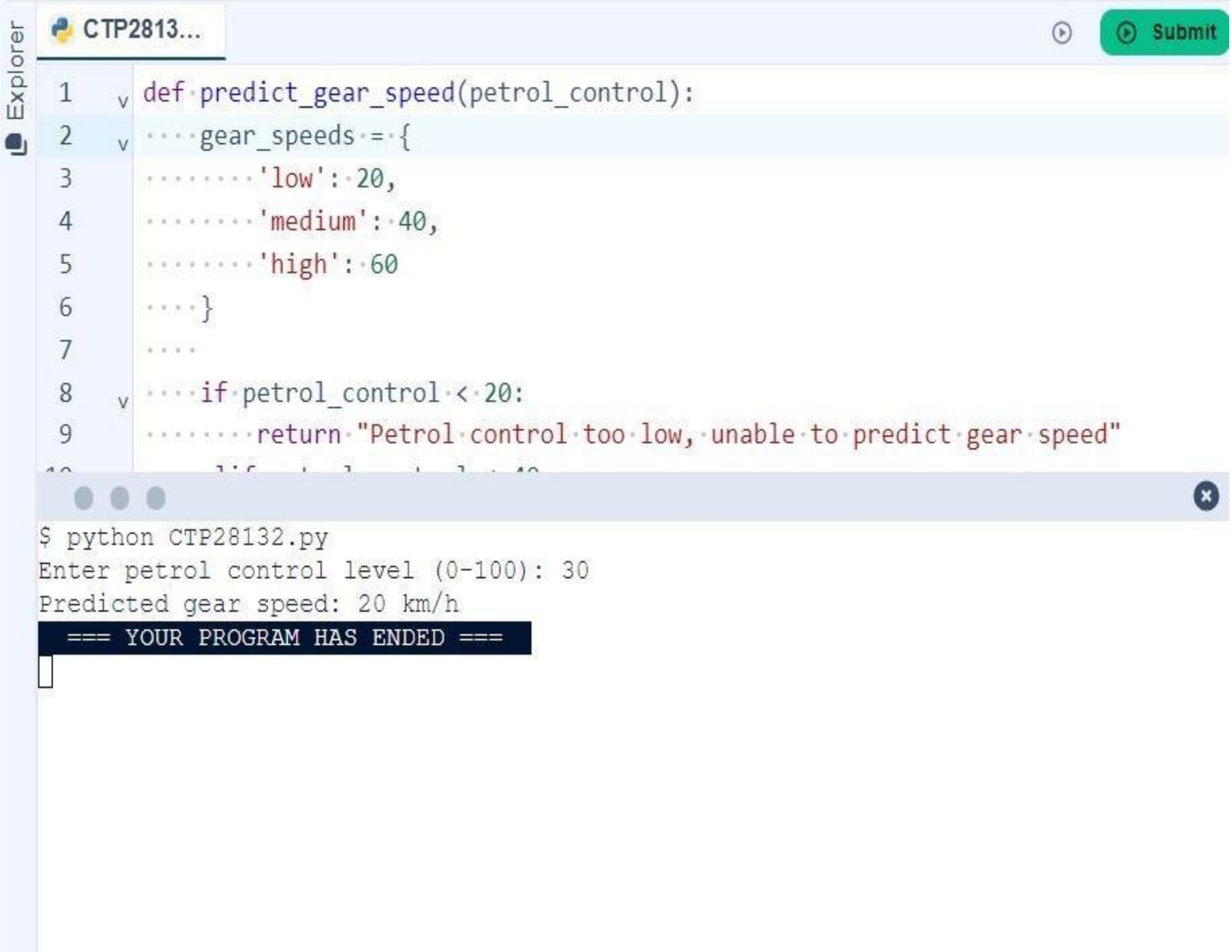
PROGRAM



```python
def predict_gear_speed(petrol_control):
    gear_speeds = {
        'low': 20,
        'medium': 40,
        'high': 60
    }
    
    if petrol_control < 20:
        return "Petrol control too low, unable to predict gear speed"
    elif petrol_control < 40:
        return gear_speeds['low']
    elif petrol_control < 60:
        return gear_speeds['medium']
    else:
        return gear_speeds['high']

# Example usage
petrol_control = int(input("Enter petrol control level (0-100): "))
predicted_speed = predict_gear_speed(petrol_control)
print("Predicted gear speed:", predicted_speed, "km/h")
```

OUTPUT



```
def predict_gear_speed(petrol_control):
    gear_speeds = {
        'low': 20,
        'medium': 40,
        'high': 60
    }
    
    if petrol_control < 20:
        return "Petrol control too low, unable to predict gear speed"
```

```
$ python CTP28132.py
Enter petrol control level (0-100): 30
Predicted gear speed: 20 km/h
 === YOUR PROGRAM HAS ENDED ===
```

CHAPTER 5

CONCLUSION

In conclusion, **PyQt5** provides a robust and flexible framework for developing high-quality, cross-platform desktop applications with graphical user interfaces (GUIs). By leveraging the extensive set of widgets, layout management tools, and event-handling mechanisms, developers can create user-friendly and interactive applications for various use cases. Its integration with Python makes it accessible to a wide range of developers, from beginners to professionals.

PyQt5's support for modern GUI design, customizability, and cross-platform compatibility ensures that applications built using this framework are both functional and visually appealing. Whether you are creating a simple desktop tool or a complex application, PyQt5 offers the necessary features and flexibility to bring your ideas to life. Its ability to handle graphics, file operations, and responsive design further enhances its suitability for diverse application development needs.

Overall, PyQt5 is a powerful and reliable tool that opens up numerous possibilities for creating intuitive, scalable, and cross-platform desktop applications.

REFERENCES:

1. Davis, S. (2020). Mastering PyQt5: Building Python GUIs. Packt Publishing.
2. Frerichs, M. (2019). PyQt5 By Example. Packt Publishing.
3. Python Software Foundation. (2024). Python 3 Documentation. Retrieved from https://docs.python.org/3/
4. Qt Documentation. (n.d.). Qt for Python - PyQt5 Overview. Retrieved from https://doc.qt.io/qtforpython/
5. Wang, H., & Zhang, X. (2021). Developing Cross-Platform Applications with PyQt5. Springer.
6. PyQt5 Documentation. (n.d.). PyQt5 Reference Guide. Retrieved from https://riverbankcomputing.com/software/pyqt/intro

APPENDIX

(Coding)

```
# Importing necessary libraries
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton,
QMessageBox

# Define the main window class
class AppWindow(QWidget):
    def __init__(self):
        super().__init__()

        # Set up the window
        self.setWindowTitle('PyQt5 Simple App')
        self.setGeometry(100, 100, 300, 200)  # Position x, y and size width, height

        # Add a button widget
        self.button = QPushButton('Click Me', self)
        self.button.setGeometry(100, 80, 100, 40)  # Button position and size
        self.button.clicked.connect(self.show_message)  # Connect button click to
function

    # Function to show a message box when the button is clicked
    def show_message(self):
        msg = QMessageBox()
        msg.setIcon(QMessageBox.Information)
        msg.setText("Hello! This is a PyQt5 message box.")
        msg.setWindowTitle("PyQt5 Message")
        msg.exec_()

# Main function to run the application
def main():
    app = QApplication(sys.argv)  # Create the application object
    window = AppWindow()  # Create an instance of the application window
    window.show()  # Display the window
    sys.exit(app.exec_())  # Execute the application

# Run the application
```

```
if __name__ == '__main__':
    main()
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QVBoxLayout,
QFormLayout, QLabel, QLineEdit, QPushButton, QComboBox

class FormWindow(QWidget):
    def __init__(self):
        super().__init__()

        self.setWindowTitle('PyQt5 Form Layout')
        self.setGeometry(100, 100, 400, 300)

        # Set up the main layout for the window
        self.layout = QVBoxLayout()

        # Create a form layout for labels and text inputs
        self.form_layout = QFormLayout()

        # Add labels and text fields
        self.name_input = QLineEdit(self)
        self.age_input = QLineEdit(self)
        self.email_input = QLineEdit(self)

        # Add the input fields to the form layout
        self.form_layout.addRow(QLabel('Name: '), self.name_input)
        self.form_layout.addRow(QLabel('Age: '), self.age_input)
        self.form_layout.addRow(QLabel('Email: '), self.email_input)

        # Create a combo box with options
        self.category_box = QComboBox(self)
        self.category_box.addItems(['Select Category', 'Food', 'Entertainment',
'Transport'])
        self.form_layout.addRow(QLabel('Category: '), self.category_box)

        # Add the form layout to the main window layout
        self.layout.addLayout(self.form_layout)

        # Add a submit button
        self.submit_button = QPushButton('Submit', self)
        self.submit_button.clicked.connect(self.submit_data)

        # Add button to layout
    self.layout.addWidget(self.submit_button)

        self.setLayout(self.layout)

    def submit_data(self):
        # Get the values from the input fields
        name = self.name_input.text()
        age = self.age_input.text()
        email = self.email_input.text()
        category = self.category_box.currentText()

        # Display the collected data
        print(f"Name: {name}, Age: {age}, Email: {email}, Category: {category}")
        self.clear_inputs()

    def clear_inputs(self):
        # Clear the input fields after submission
        self.name_input.clear()
        self.age_input.clear()
        self.email_input.clear()
        self.category_box.setCurrentIndex(0)  # Reset to default option

def main():
    app = QApplication(sys.argv)
    window = FormWindow()
    window.show()
    sys.exit(app.exec_())

if __name__ == '__main__':
    main()
```