

Project Cost :

Sl. No.	Components	Quantity	Cost	Total
1	Arduino Uno	1	800	800
2	HC-05 module	1	200	200
3	Motor Driver	1	160	160
4	Gear Motor	2	70	140
5	Wheel	2	35	70
6	9V Battery	1	20	20
7	Jumper Wire	2	20	40
8	switch	1	5	5
9	Bread Board	1	60	60
10	others	--	--	40
			Total	1535

Introduction :

The GUI-controlled car project is an exciting venture that combines the power of Arduino microcontrollers and graphical user interfaces (GUI) to control a small car wirelessly. By leveraging the Arduino platform, we can create a customizable and interactive experience that allows users to remotely maneuver the car using graphical user interface.

The project involves building a small car chassis equipped with an Arduino board, motor drivers, wheels, and other necessary components. The Arduino acts as the brain of the car, processing commands received from the GUI interface and translating them into motor movements to control the car's direction and speed.

Once the hardware is assembled and the GUI is developed, the user can interact with the GUI to control the Arduino car. The GUI can provide various controls like directional buttons, speed sliders, or even an accelerometer-based control system. As the user interacts with the GUI elements, the corresponding commands are sent wirelessly to the Arduino, which then translates them into motor movements, allowing the car to move forward, backward, turn left, turn right, and adjust its speed.

Key Components:

1. **Arduino Board:** An Arduino board, such as Arduino Uno or Arduino Nano, serves as the central processing unit of the car. It receives commands from the GUI and controls the motor drivers accordingly.
2. **Motor Drivers:** Motor drivers, such as H-bridges or motor driver shields, are used to interface between the Arduino and the car's motors. They enable controlling the direction and speed of the motors based on the commands received.
3. **Motors and Wheels:** The car is equipped with motors and wheels that provide the necessary propulsion and movement. The motors are connected to the motor drivers, which regulate their speed and direction.
4. **Wireless Communication:** To establish communication between the GUI and the Arduino, a wireless module like Bluetooth is employed. This module allows transmitting commands from the GUI to the Arduino wirelessly, providing convenient control over the car. We used HC-05 bluetooth module for this project .
5. **Graphical User Interface (GUI):** The GUI serves as the control interface for the project. It can be developed using programming languages like Python, C#, or Java, and it provides buttons, sliders, or other interactive elements to send commands to the Arduino wirelessly. We use python programming language for creating this project gui.
6. **Others :** Usb cable , Battery , Breadboard, Glue gun and jumper wire.

Python :

Python is a high-level, interpreted programming language known for its simplicity and readability. It was created by Guido van Rossum and first released in 1991. Python's design philosophy emphasizes code readability and a clean syntax, making it easier to write and understand compared to other programming languages. It used for –

- Web Development (server-side)
- software development
- Mathematics
- system scripting
- Game Development etc.

Key features of Python include:

- 1.**Readability:** Python code is designed to be highly readable and expressive, using indentation and a minimalistic syntax. This makes it easier to write and maintain code, enhancing collaboration among developers.
- 2.**Interpreted:** Python is an interpreted language, which means that code is executed line by line without the need for compilation. This allows for quick prototyping and interactive development.
- 3.**Object-Oriented:** Python supports object-oriented programming (OOP) principles, allowing developers to create reusable and modular code through the use of classes, objects, and inheritance.
- 4.**Dynamic Typing:** Python is dynamically typed, meaning that variable types are determined automatically at runtime. This flexibility allows for more concise code and faster development iterations.

GUI :

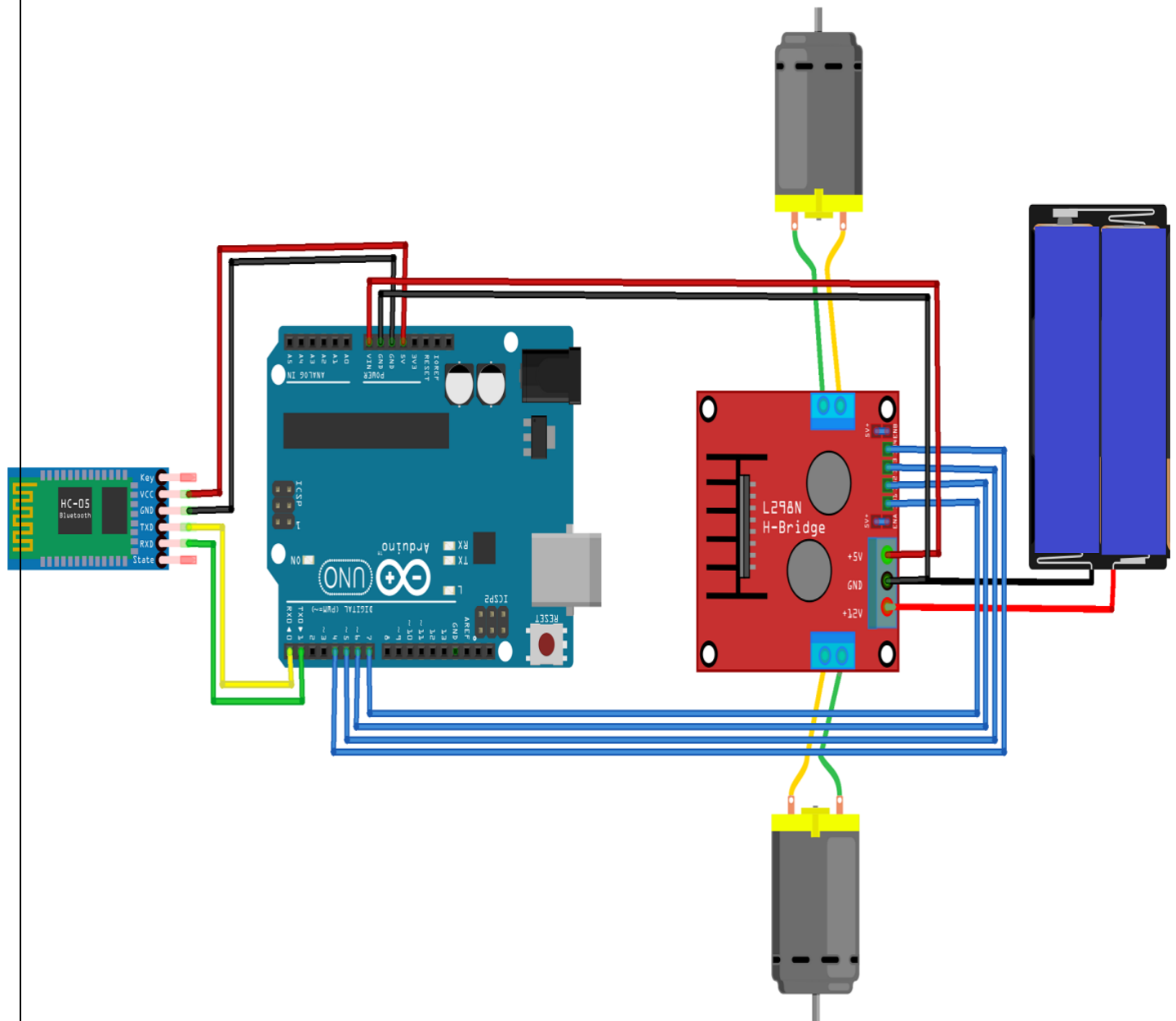
GUI stands for Graphical User Interface. It refers to the visual elements and interactive components that allow users to interact with software or computer systems. A GUI provides a graphical representation of the system, enabling users to perform actions and receive feedback through visual elements like windows, icons, menus, buttons, and forms.

In a GUI, users can interact with the software by using a mouse, touchpad, or other input devices. They can click on buttons, select options from menus, drag and drop elements, input data into forms, and navigate through various screens or windows.

GUIs provide a user-friendly and intuitive way to interact with software applications, allowing users to perform complex tasks without needing to remember or type complex commands. The visual representation and interactive elements make it easier for users to understand the system's functionality and provide input or receive output.

Examples of GUIs are desktop operating systems like Windows, macOS, and Linux, which provide a graphical interface for users to interact with files, applications, and settings. GUIs are also prevalent in various software applications, such as web browsers, image editors, word processors, spreadsheets, and many other types of software.

Arduino Setup :




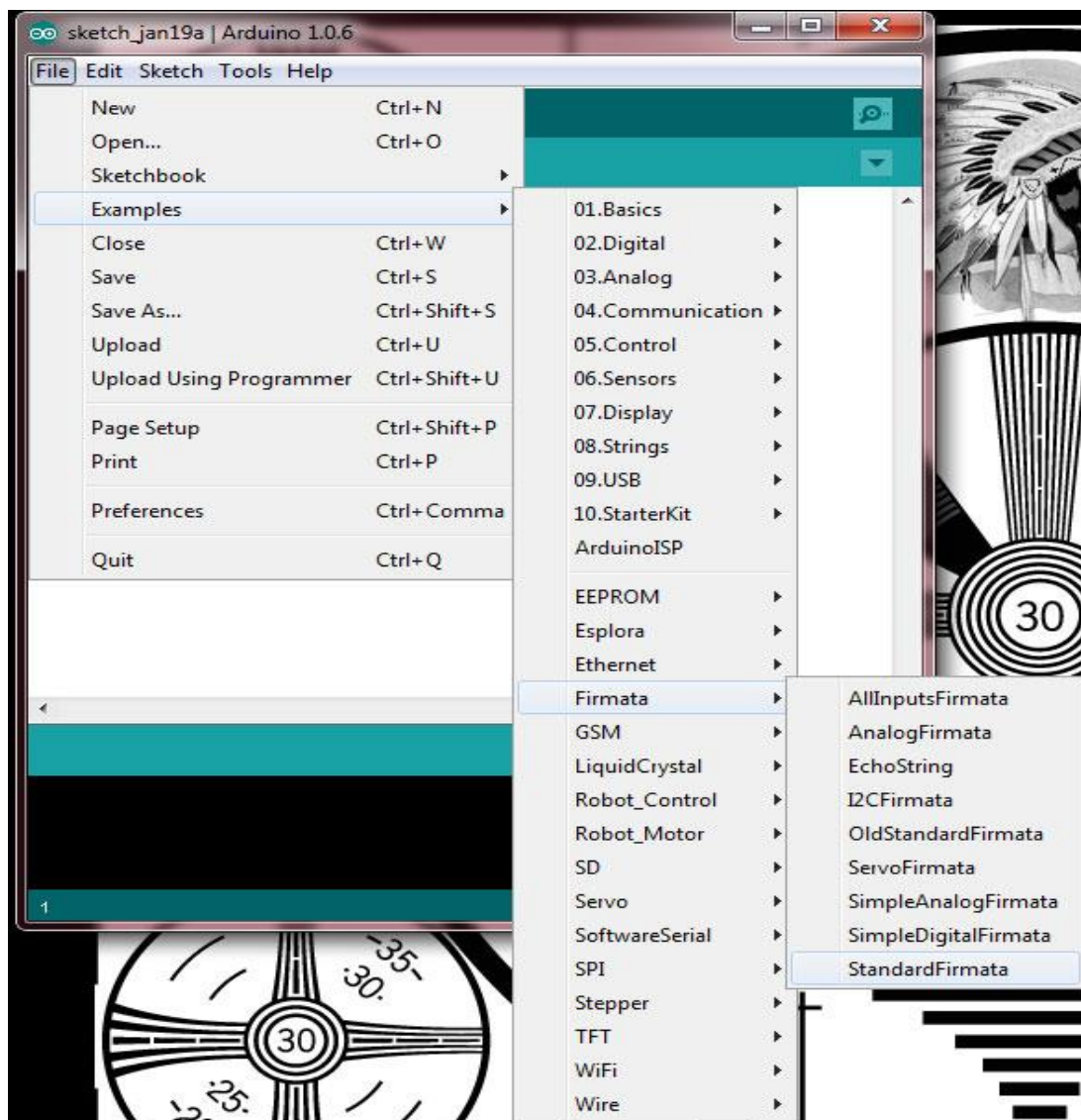
Arduino Code:

We uplode Firmata library on Arduin board.

Firmata : Firmata is a protocol and firmware that allows for easy communication between software running on a computer and Arduino microcontrollers.

Steps to upload Firmata :

1. Open Arduino IDE and connect arduino board using USB.
2. Select board and port from tool section.
3. Go to file section > Example > Firmata > StandardFirmata
4. Then upload this code using  on from the top button section.



Python Code :

```
from pyfirmata import Arduino

import time

from pynput.keyboard import Key , Listener

board = Arduino('COM3')

pin_3 = board.get_pin('d:3:p')

pin_6 = board.get_pin('d:6:p')

pin_2 = board.get_pin('d:2:o')

pin_4 = board.get_pin('d:4:o')

pin_5 = board.get_pin('d:5:o')

pin_7 = board.get_pin('d:7:o')

speed = 0.6

def forward():

    global speed

    pin_3.write(speed)

    pin_5.write(0)

    pin_7.write(1)

    pin_6.write(speed)

    pin_2.write(0)

    pin_4.write(1)

def stop():

    pin_3.write(0)

    pin_5.write(0)

    pin_7.write(0)

    pin_6.write(0)

    pin_2.write(0)

    pin_4.write(0)
```

```
def reverse():

    global speed

    pin_6.write(speed)

    pin_2.write(1)

    pin_4.write(0)

    pin_3.write(speed)

    pin_5.write(1)

    pin_7.write(0)

def turn_right():

    global speed

    pin_6.write(speed)

    pin_2.write(1)

    pin_4.write(0)

    pin_3.write(speed)

    pin_5.write(0)

    pin_7.write(1)

def turn_left():

    global speed

    pin_6.write(speed)

    pin_2.write(0)

    pin_4.write(1)

    pin_3.write(speed)

    pin_5.write(1)

    pin_7.write(0)

def start_car():

    app.destroy()

def exit():

    app.destroy()
```


GUI Code :

```
import tkinter

import threading

import customtkinter

customtkinter.set_appearance_mode('System')

customtkinter.set_default_color_theme('blue')


app= customtkinter.CTk()

app.geometry("700x450")

app.title("Remote Gui")


titel= customtkinter.CTkLabel(app,text="Control Section")

titel.pack(padx=10, pady=10)

button1 = customtkinter.CTkButton(app,text="Start" , command= start_car )

button1.pack(padx=2, pady=2)

button2 = customtkinter.CTkButton(app, text="Stop" , command= stop)

button2.pack()

button3 = customtkinter.CTkButton(app, text="Exit" , command= exit)

button3.pack( padx=20, pady=20)

button4 = customtkinter.CTkButton(app, text="Forward" , command= forward)

button4.pack(padx=50, pady=30)

button5 = customtkinter.CTkButton(app, text="Left" , command= turn_left)

button5.pack(side=customtkinter.LEFT, padx=50, pady=10)

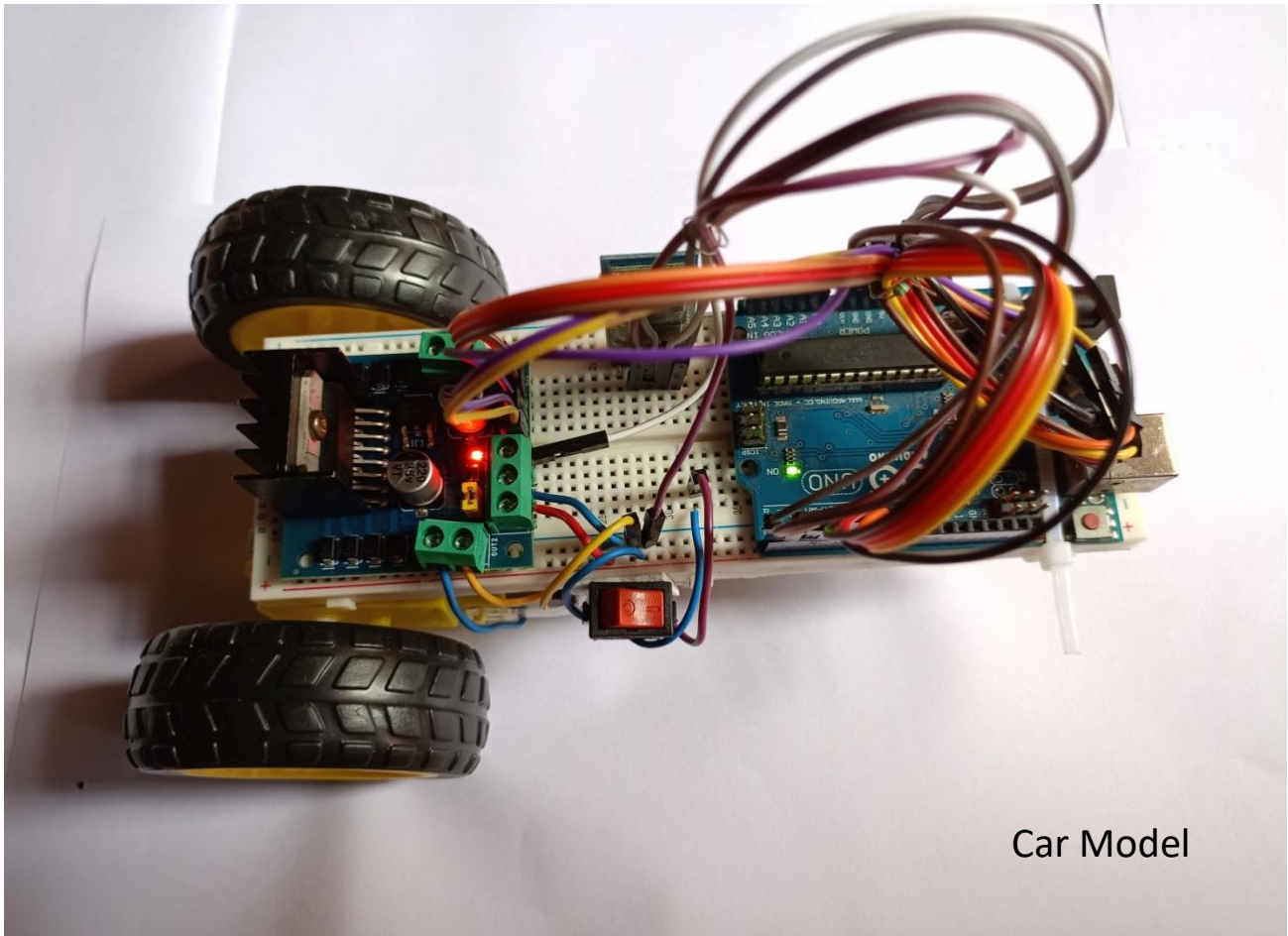
button6 = customtkinter.CTkButton(app, text="Right" , command= turn_right)

button6.pack(side=customtkinter.RIGHT, padx=50, pady=10)

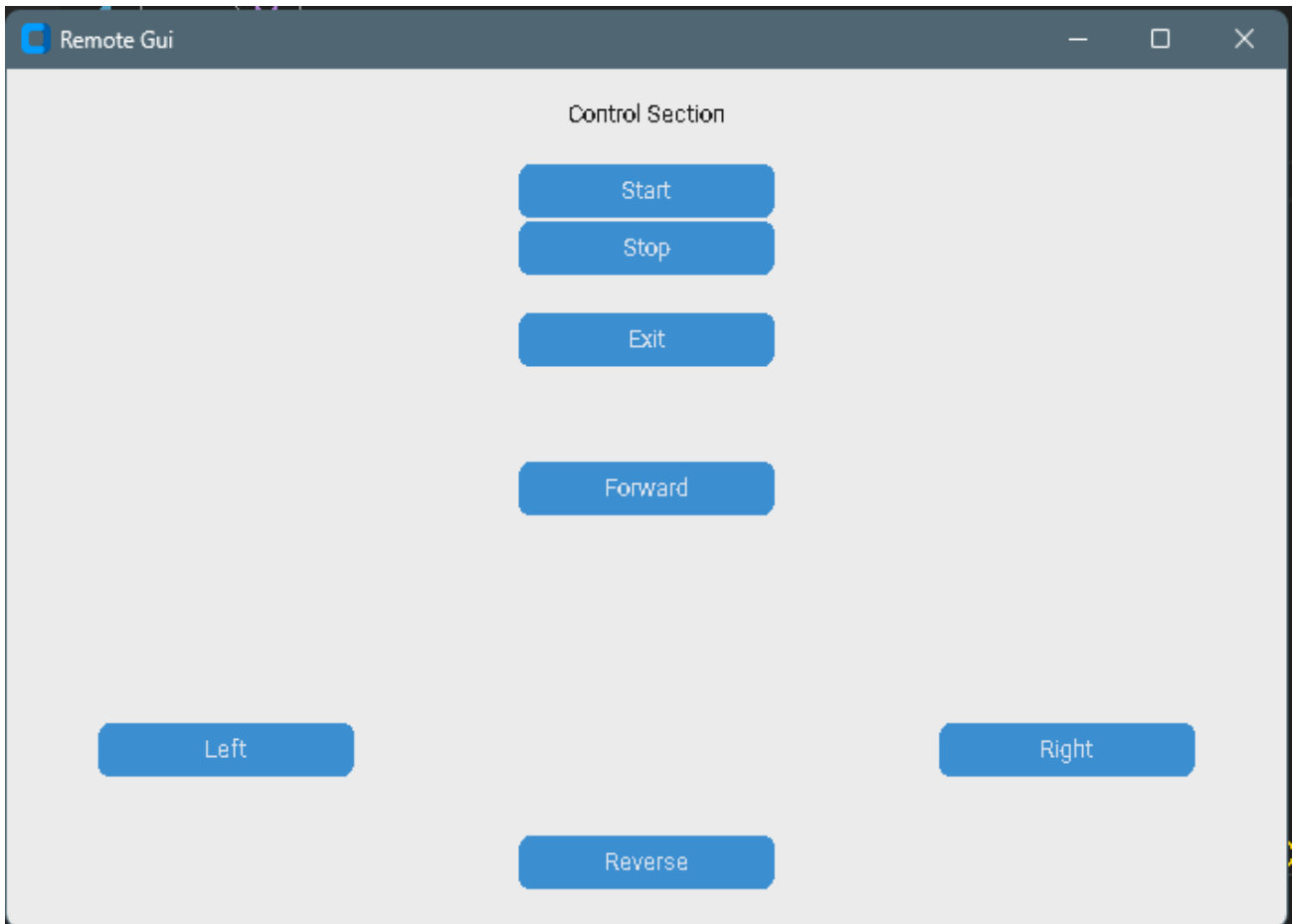
button7 = customtkinter.CTkButton(app, text="Reverse" , command= reverse)

button7.pack(side=customtkinter.BOTTOM,padx=20, pady=20)


app.mainloop()
```



Car Model



GUI Design

Conclusion:

In conclusion, the GUI-controlled Arduino car project offers an exciting opportunity to delve into the realms of robotics, programming, and user interface design. By combining Arduino microcontrollers, motor drivers, wireless communication, and a graphical user interface (GUI), this project provides a customizable and interactive experience.

Throughout the project, you learned how to assemble a small car chassis equipped with an Arduino board, motor drivers, and wheels. You explored the process of uploading the Firmata firmware to the Arduino, establishing a communication channel between the Arduino and the GUI.

The GUI served as the control interface, allowing users to remotely maneuver the car using a computer or mobile device. With the GUI, you provided intuitive controls such as directional buttons, speed sliders, or even accelerometer-based control, enhancing the user experience.

Reference :

1. Arduino Forum: The Arduino Forum (<https://forum.arduino.cc/>) is a community platform. Which is valuable resource for finding project ideas, troubleshooting, and seeking guidance.
2. GitHub: GitHub (<https://github.com/>) , Which is a code repository and collaboration platform that hosts numerous Arduino projects and GUI development.
3. Python Subreddit and Python Discord: The Python subreddit (<https://www.reddit.com/r/python/>) and Python Discord community (<https://discord.gg/python>)
4. Online Tutorials and Project Blogs: Hackster.io (<https://www.hackster.io/>), Instructables (<https://www.instructables.com/>)