**Practice Simulation of Pushbutton Control for LED and Relay on ESP32**

*Andreika Luna Alghivari*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: andreikaluna@student.ub.ac.id*

**Abstract**

This study focuses on the simulation practice of integrating a relay, button, and LED with an ESP32 microcontroller using the Wokwi platform and PlatformIO in Visual Studio Code. The simulation demonstrates how to control a relay and LED through button inputs, showcasing the ESP32's GPIO capabilities and its application in automation systems. By leveraging Wokwi's virtual environment, this project eliminates the need for physical hardware, providing a cost-effective and accessible way to learn about microcontroller-based control systems. The successful implementation highlights its potential for real-world applications such as smart home automation and device control.

*Keywords: GPIO, ESP32, Button, LED, Relay, Wokwi, Visual Studio Code, PlatformIO, Arduino.*

**1. Introduction**

* 1. **Background**

The ESP32 microcontroller is widely used in IoT and embedded systems due to its robust features, including Wi-Fi connectivity and versatile GPIO pins. Relays are essential for controlling high-power devices using low-power signals from microcontrollers, while LEDs provide visual feedback, and buttons serve as user input devices. Simulating these components on the Wokwi platform allows developers to design and test circuits virtually without physical hardware. Using PlatformIO in Visual Studio Code provides an efficient development environment with advanced features like debugging and code management. This combination of tools offers an ideal setup for learning and prototyping IoT-based control systems.

* 1. **Objective**

1. To demonstrate the integration of a relay, button, and LED with an ESP32 microcontroller using the Wokwi simulator.
2. To program the ESP32 through PlatformIO in Visual Studio Code for controlling the relay and LED based on button inputs.

**2. Methodology**

**2.1 Tools & Materials**

LED, Relay module, Pushbutton, Wokwi, ESP32, Arduino IDE, and Visual Studio Code.

**2.2 Implementation Steps**

1. Create a new project in wokwi and select ESP32
2. Add the relay, pushbutton, and LED. Connect the pushbutton to GPIO19 using an internal pull-up configuration, the LED to GPIO18, and the relay to GPIO23.
3. Enter the sketch.ino code according to the module
4. Create a new project in platform.io
5. Enter the C++ code in src/main.cpp
6. Edit the platform.ini file according to the module
7. Create a diagram.json file and copy and paste the code from diagram.json into wokwi
8. Create a wokwi.toml file and fill in the firmware code and elf from the copy relative path after compiling the main.cpp file
9. Request license to wokwi.com
10. Start Simulator

**3. Results and Discussion**

**3.1 Experimental Results**

The simulation of integrating a relay, button, and LED with the ESP32 using the Wokwi platform and PlatformIO in Visual Studio Code was successfully executed. The system effectively controlled the relay and LED based on button inputs, demonstrating the ESP32's capabilities in automation. This virtual setup provided an accessible way to learn microcontroller-based control systems without physical hardware. The success of this project highlights its potential for real-world applications, such as smart home automation and IoT solutions.

1. Diagram.json



1. Main.cpp

A screenshot of a computer program

Description automatically generated

1. Results

A screenshot of a computer

Description automatically generated

**Practice Simulation Ultrasonic Sensor Based Distance Measurement Using ESP32**

*Andreika Luna Alghivari*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: andreikaluna@student.ub.ac.id*

**Abstract**

This project demonstrates the simulation of an HC-SR04 ultrasonic distance sensor integrated with an ESP32 microcontroller using the Wokwi platform and Visual Studio Code. The HC-SR04 sensor calculates distances by emitting ultrasonic waves and measuring the time taken for the echo to return. The ESP32 processes this data and displays the measured distance in real-time. This simulation provides a virtual environment for learning sensor integration, programming, and data processing, making it ideal for IoT prototyping. The setup highlights practical applications such as obstacle detection, smart devices, and automation systems.

*Keywords: Ultrasonic Sensor, ESP32, HC-SR04, Wokwi, Visual Studio Code, PlatformIO, Arduino.*

**1. Introduction**

* 1. **Background**

The HC-SR04 ultrasonic sensor is widely used for contactless distance measurement in a range of 2 cm to 400 cm. It operates by transmitting ultrasonic waves via its trigger pin and receiving the reflected waves through its echo pin. The ESP32 microcontroller, known for its advanced processing power and wireless connectivity, is ideal for interfacing with sensors like the HC-SR04. Using the Wokwi simulator eliminates the need for physical hardware, enabling developers to design and test their projects virtually. Visual Studio Code with PlatformIO enhances this process by offering an efficient coding environment with debugging capabilities.

* 1. **Objective**

1. To integrate the HC-SR04 ultrasonic sensor with an ESP32 microcontroller using Wokwi.
2. To program the ESP32 to calculate and display real-time distance measurements based on ultrasonic sensor data.
3. To provide hands-on experience in designing IoT systems using a virtual platform.

**2. Methodology**

**2.1 Tools & Materials**

HC-SR04 Ultrasonic Sensor, Wokwi, ESP32, Arduino IDE, and Visual Studio Code.

**2.2 Implementation Steps**

1. Create a new project in wokwi and select ESP32
2. Add HC-SR04 and set VCC, ECHO, TRIG, and GND
3. Enter the sketch.ino code according to the module
4. Create a new project in platform.io
5. Enter the C++ code in src/main.cpp
6. Edit the platform.ini file according to the module
7. Create a diagram.json file and copy and paste the code from diagram.json into wokwi
8. Create a wokwi.toml file and fill in the firmware code and elf from the copy relative path after compiling the main.cpp file
9. Request license to wokwi.com
10. Start Simulator

**3. Results and Discussion**

**3.1 Experimental Results**

The simulation of the HC-SR04 ultrasonic distance sensor with the ESP32 microcontroller using the Wokwi platform was successfully executed. The system accurately measured and displayed distance readings in real-time, demonstrating effective integration and data processing capabilities. This successful implementation validated the functionality of both the ultrasonic sensor and the ESP32, showcasing their potential in various IoT applications. The virtual environment provided by Wokwi allowed for a seamless learning experience without the need for physical hardware. Overall, this project highlights the practical use of ultrasonic sensors in applications such as obstacle detection and smart automation systems, paving the way for further exploration and development in IoT solutions.

1. Diagram.json

A screenshot of a computer

Description automatically generated

1. Main.cpp

A screen shot of a computer program

Description automatically generated

1. Results

A screenshot of a computer

Description automatically generated

**Practice Making API Using Laravel 11 and Ngrok**

*Andreika Luna Alghivari*

*Fakultas Vokasi, Universitas Brawijaya*

*Email: andreikaluna@student.ub.ac.id*

**Abstract**

This project explores the practice of creating and testing RESTful APIs using Laravel 11 and Ngrok on the Wokwi platform within Visual Studio Code. Laravel, a robust PHP framework, simplifies the development of APIs with its built-in features, while Ngrok enables secure tunneling for local APIs to be accessed globally. The simulation demonstrates the step-by-step process of building scalable APIs, including CRUD operations, authentication, and endpoint testing. This practice provides a comprehensive understanding of API development and deployment, paving the way for real-world applications in web and mobile systems.

*Keywords: Laravel, API, Ngrok, Postman, Visual Studio Code.*

**1. Introduction**

* 1. **Background**

Laravel is one of the most popular PHP frameworks due to its elegant syntax, built-in tools for API development, and support for modern web applications. RESTful APIs facilitate communication between systems using standard HTTP methods like GET, POST, PUT, and DELETE. Ngrok complements this by creating secure tunnels that expose local APIs to the internet, enabling remote access for testing or integration purposes. Visual Studio Code with PlatformIO serves as an efficient development environment for managing code and debugging. This combination of tools offers developers a streamlined workflow for building robust APIs while ensuring accessibility during testing.

* 1. **Objective**

1. To demonstrate the creation of a RESTful API using Laravel 11
2. To integrate Ngrok for exposing local APIs to the internet securely.
3. To test API endpoints using tools like Postman or cURL for validating functionality.

**2. Methodology**

**2.1 Tools & Materials**

Laravel 11, Postman, Xampp, Ngrok, Composer, and Visual Studio Code.

**2.2 Implementation Steps**

1. Download the packages needed for laravel 11 with the command in the terminal

composer create-project --prefer-dist laravel/laravel:11.0 laravel-11

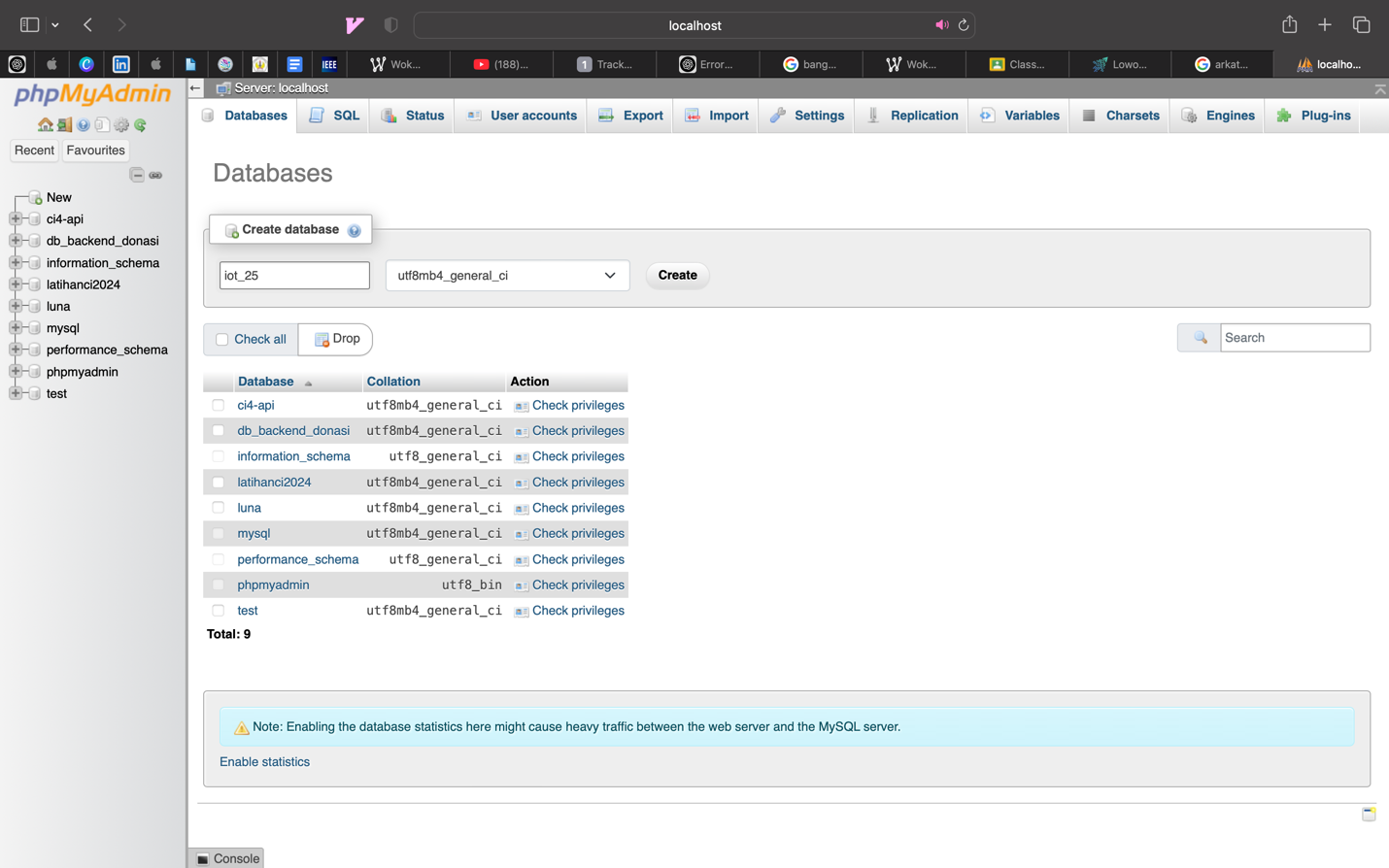
cd laravel-11

1. Create a database on phpmyadmin with the name iot\_25
2. Change the contents of the .env file configuration in the db\_username and db\_password sections according to the laptop settings
3. Create the TransactionSensor.php model file (php artisan make:model TransactionSensor -m) then change the create transaction sensors table file in the databases-migrations folder
4. Change the contents of the app/Models/TransactionSensor.php file
5. Create the table by running the php artisan migrate command on the terminal
6. Create a resource by running the command php artisan make:resource TransactionSensorResource
7. Change the contents of the TransactionSensorResource.php file in the app-Http-Resources folder
8. Create an API controller with the command php artisan make:controller Api/TransactionSensorController, then change the contents of the file
9. Create a special API route with the command php artisan install:api, then change the contents of the routes/api.php file
10. Ensure that the routes have been formed with the php artisan route:list command
11. Testing with the postman application, in the url section enter the laravel servel address http://localhost:8000/api/posts
12. Select the GET method to retrieve data from the database, then click the SEND button
13. Then try to insert data into a table in the database using the API
14. Check manually in phpmyadmin and make sure the new data enters
15. Connect the API using the ngrok service so that the API can be accessed via an iot device or wokwi iot simulation by running the ngrok command http http://localhost:8000
16. Experiment using postman using the URL given by ngrok. For GET API experiments, the URL must add the endpoint address /api/posts

**3. Results and Discussion**

**3.1 Experimental Results**

1. Make database on phpMyAdmin



1. Make folder “latihanbab12”

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Create the TransactionSensor.php model file (php artisan make:model TransactionSensor)

A screenshot of a computer

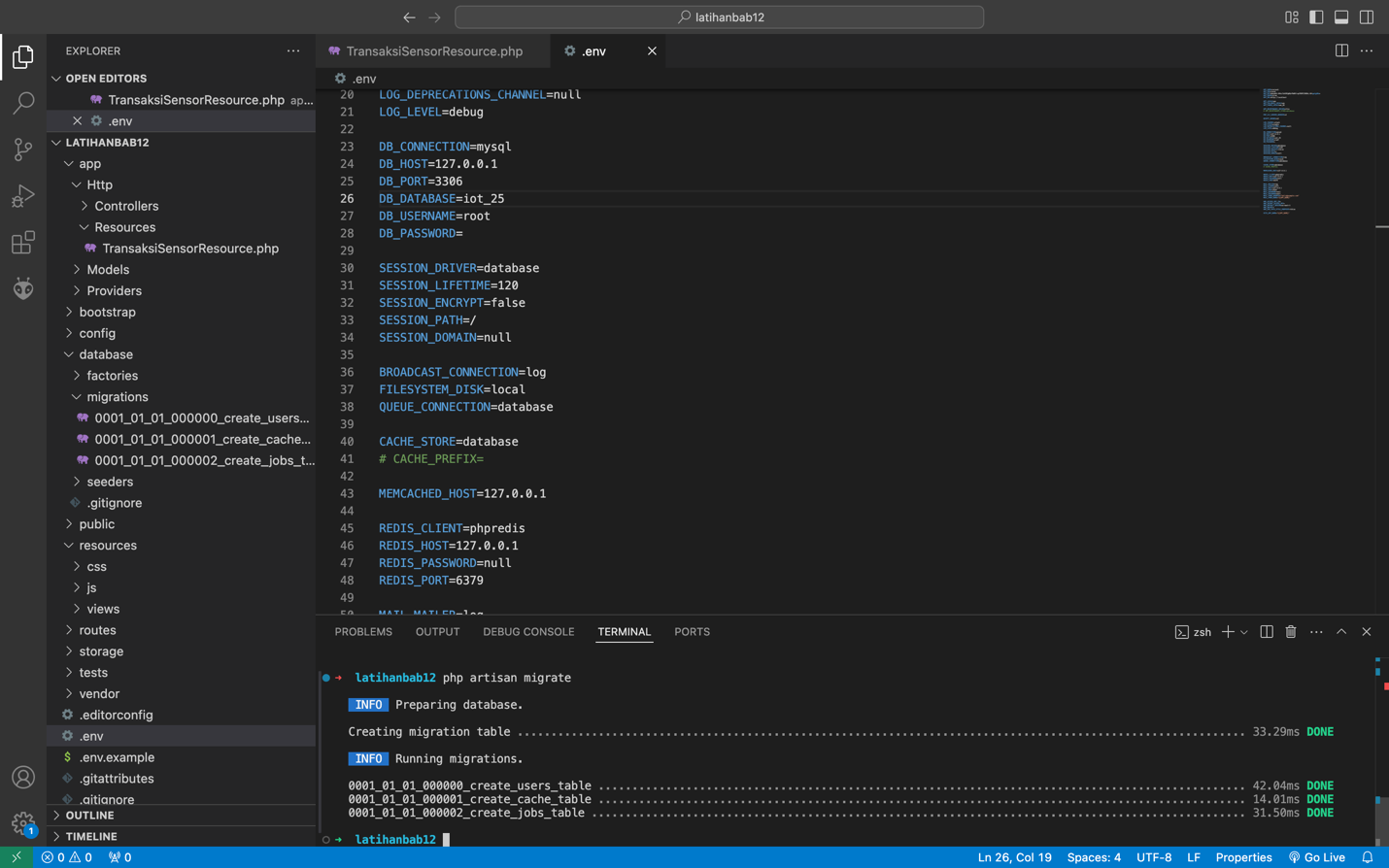
Description automatically generated

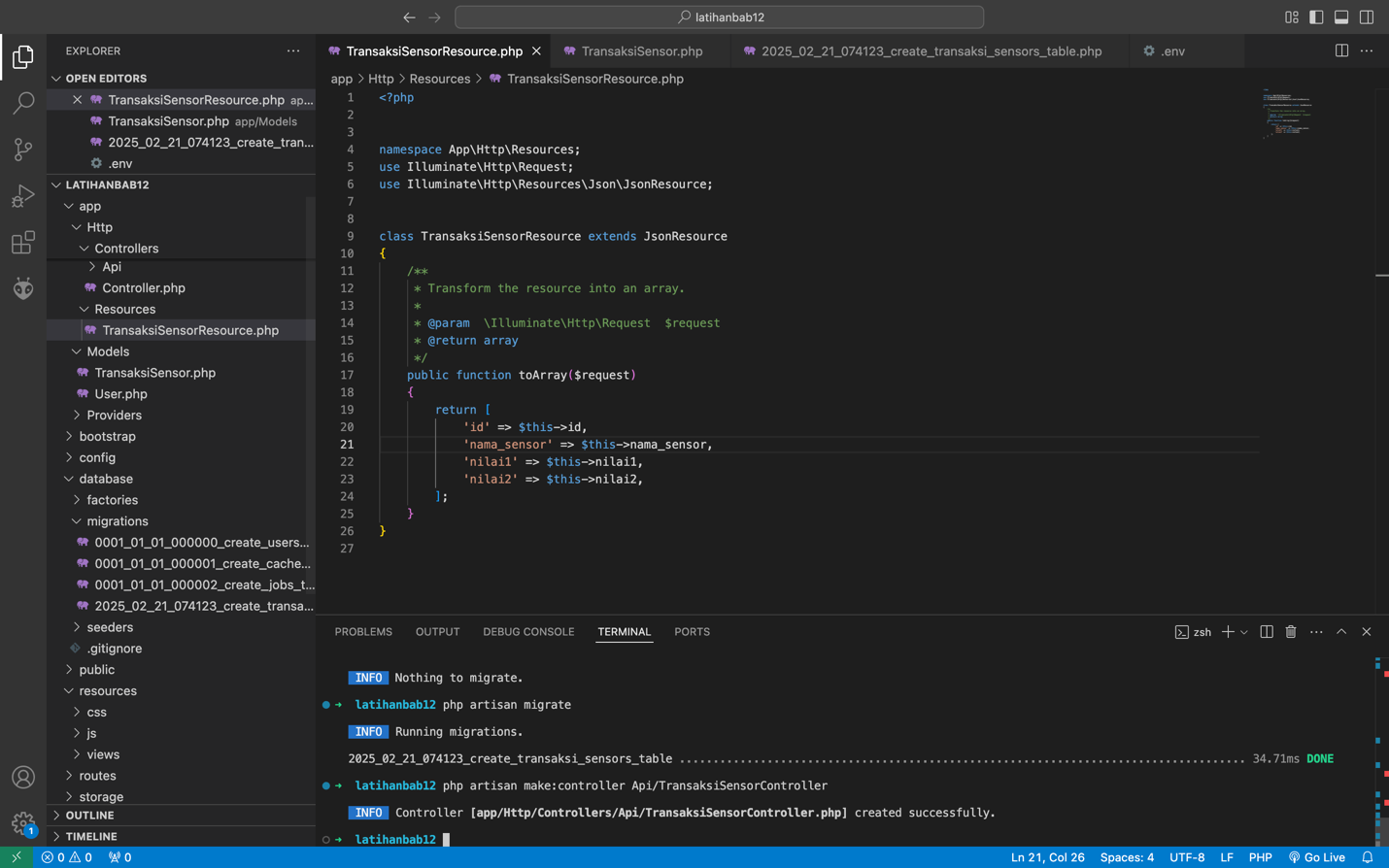
1. Create a resource by running the command php artisan make:resource TransactionSensorResource

A screenshot of a computer

Description automatically generated

1. Then Migrate





1. Testing with the postman application

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Check manually in phpmyadmin and make sure the new data enters

A screenshot of a computer

Description automatically generated

1. Connect the API using the ngrok service so that the API can be accessed via an iot device or wokwi iot simulation by running the ngrok command http <http://localhost:8000>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. Experiment using postman using the URL given by ngrok. For GET API experiments, the URL must add the endpoint address /api/posts

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

At this stage, the API developed using Laravel 11 is functioning effectively and can be accessed through a public URL. The successful implementation of CRUD operations and authentication mechanisms demonstrates the robustness of the API. Utilizing Ngrok has enabled secure tunneling, allowing for remote access to the local development environment. This accomplishment not only validates the development process but also highlights the potential for deploying the API in real-world applications. Moving forward, this API can serve as a foundational component for various web and mobile applications, facilitating seamless integration and communication between systems.