CHAPTER 4
RESULTS AND DISCUSSION

This chapter contains the test results of the system that has been designed in the previous chapter. Testing begins by ensuring that every component of the hardware and software can work according to the previous design, after testing the hardware and software, then testing the encryption and decryption of data.

4.1 Hardware Realization

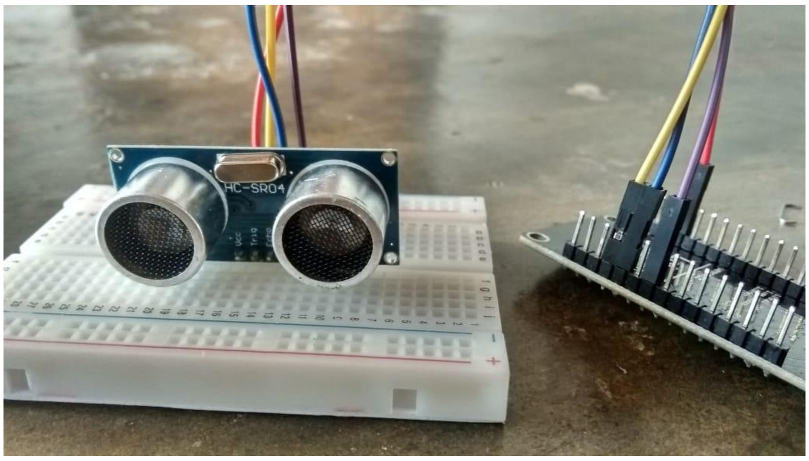


Figure 4.1 Hardware Realization

The main hardware components are the HC-SR04 sensor module and the NodeMCU ESP8266 microcontroller. The HC-SR04 sensor module is connected to the NodeMCU microcontroller via pins using jumper cables. The Trig pin is connected to the D1 pin of the NodeMCU, the Echo pin is connected to the D0 pin of the NodeMCU, then the Vcc pin is connected to the Vin pin on the NodeMCU and the Gnd pin is connected to the G pin on the NodeMCU. NodeMCU is connected to a laptop using a USB cable as a power source and can upload program code. The client and server are connected via an internet connection from a smartphone (tethering).

4.2 Software Realization

The following is the realization of the design of the client and server:

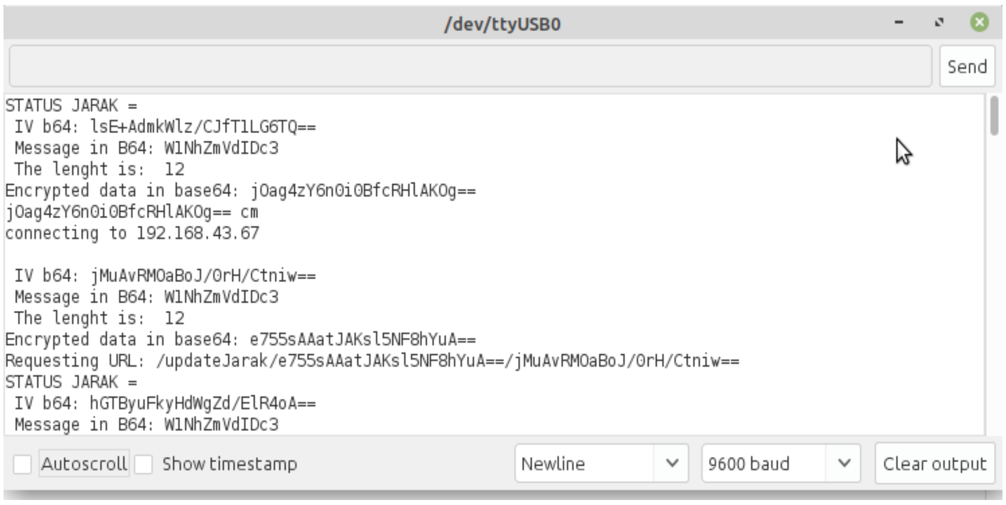
4.2.1 Software Realization (Client)

In the realization of this software, Arduino IDE is used as a tool that is used to write program code that runs on the microcontroller. It is necessary to configure several boards in the Arduino IDE in order to upload the program code correctly without errors.

Table 4.1 Arduino IDE Software Configuration

Board	NodeMCU 1.0 (ESP-12E Module)
Upload Speed	9600
Debug Port	Disable
Debug levels	None

4.2.2 Software Realization Results (Client)

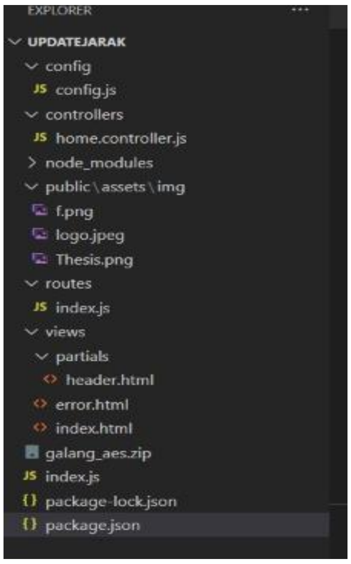


```
STATUS JARAK =
 IV b64: lsE+AdmkWlz/CJfT1LG6TQ==
 Message in B64: W1NhZmVdIDc3
 The lenght is:  12
Encrypted data in base64: jOag4zY6n0i0BfcRHlAKOg==
jOag4zY6n0i0BfcRHlAKOg== cm
connecting to 192.168.43.67

 IV b64: jMuAvRMOaBoJ/0rH/Ctniw==
 Message in B64: W1NhZmVdIDc3
 The lenght is:  12
Encrypted data in base64: e755sAAatJAKsl5NF8hYuA==
Requesting URL: /updateJarak/e755sAAatJAKsl5NF8hYuA==/jMuAvRMOaBoJ/0rH/Ctniw==
STATUS JARAK =
 IV b64: hGTByuFkyHdWgZd/ElR4oA==
 Message in B64: W1NhZmVdIDc3
```

Figure 4.2 Software Realization (Client)

Figure 4.3 shows the client system display on the Arduino IDE Serial monitor. The NodeMCU is connected to a wifi network with the IP address "192.168.43.67" then the HC-SR04 sensor reads the distance data. The distance data is then encrypted using the 128 bit AES algorithm into cipher text. Then the client sends the cipher text to the server.

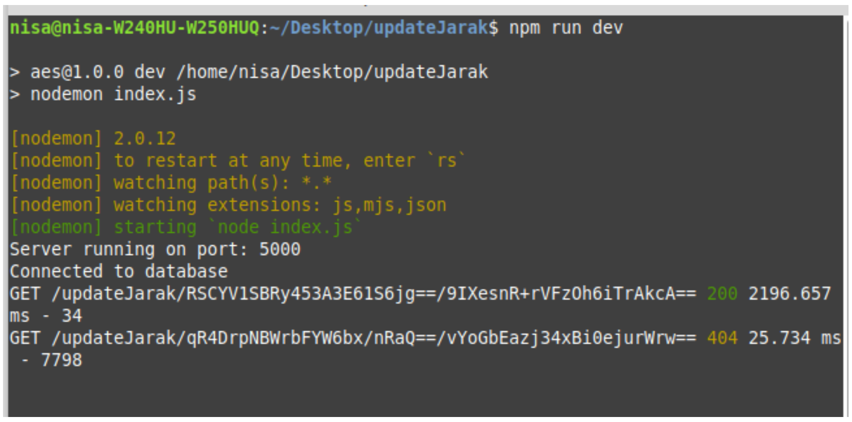
4.2.3 Software Realization (Server)



```
EXPLORER
UPDATEJARAK
  config
    config.js
  controllers
    home.controller.js
  node_modules
  public\assets\img
    f.png
    logo.jpeg
    Thesis.png
  routes
    index.js
  views
    partials
      header.html
    error.html
    index.html
  galang_aes.zip
  index.js
  package-lock.json
  package.json
```

Figure 4.3 Software Realization (Server)

In Figure 4.3 Realization of server side software using visual studio code. There are 12 files in visual studio code, these files have their respective uses both for database configuration, designing the appearance of the web server as well as coding for AES Key.

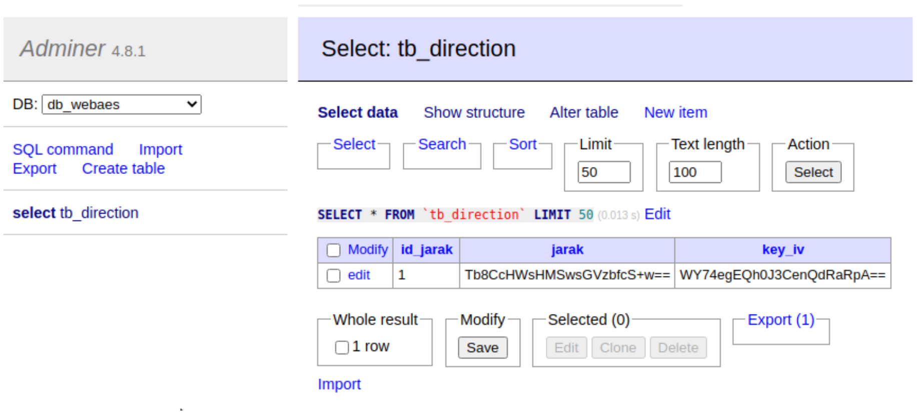


```
nisa@nisa-W240HU-W250HUQ:~/Desktop/updateJarak$ npm run dev

> aes@1.0.0 dev /home/nisa/Desktop/updateJarak
> nodemon index.js

[nodemon] 2.0.12
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Server running on port: 5000
Connected to database
GET /updateJarak/RSCYV1SBRy453A3E61S6jg==/9IXesnR+rVFzOh6iTrAkcA== 200 2196.657
ms - 34
GET /updateJarak/qR4DrpNBWrbFYW6bx/nRaQ==/vYoGbEazj34xBi0ejurWrw== 404 25.734 ms
 - 7798
```

Figure 4.4 Server Software Realization Results on Terminal

In Figure 4.4 is a display of server software results, the command "npm run dev" is used to activate nodemon 2.0.12, to show that the web service is running on port 5000 which in this case is an HTTP port and is used to connect to the database.

4.2.4 Realization of Database View


Adminer 4.8.1

DB: db_webaes

SQL command Import
Export Create table

select tb_direction

Select: tb_direction

Select data Show structure Alter table New item

Select Search Sort Limit 50 Text length 100 Action Select

```
SELECT * FROM `tb_direction` LIMIT 50 (0.013 s) Edit
```

Modify	id_jarak	jarak	key_iv
edit	1	Tb8CcHWsHMSwsGVzbfcS+w==	WY74egEQh0J3CenQdRaRpA==

Whole result: 1 row · Modify: Save · Selected (0): Edit Clone Delete · Export (1)

Import

Figure 4.5 Database View

The database design that has been done on visual studio code has a final appearance as shown in Figure 4.5. In the database view, it can be seen that the database name is "db_webaes" with the table name "tb_direction" which contains "id_jarak" with int type Auto Increment as well as "distance" and "key_iv" with type varchar(256). "distance" and "key_iv" on the display, encrypted or still in the form of cipher text.

4.3 Realization of AES Algorithm

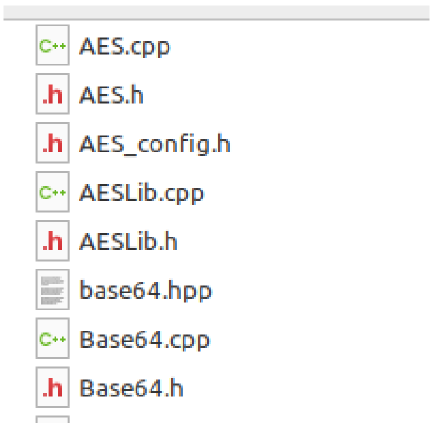
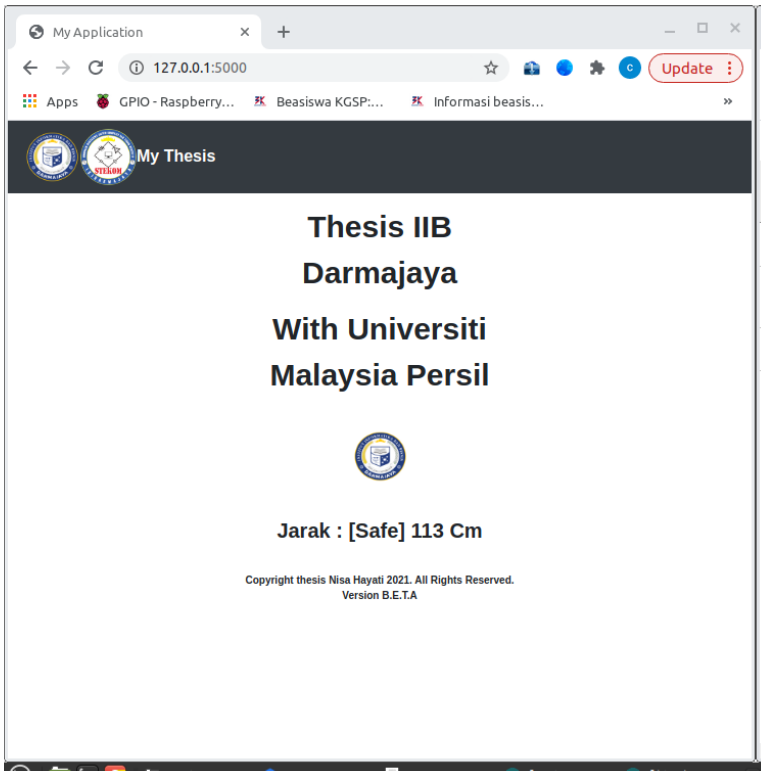

AES.cpp
AES.h
AES_config.h
AESLib.cpp
AESLib.h
base64.hpp
Base64.cpp
Base64.h

Figure 4.6 Realization of AES Algorithm

In Figure 4.6 shows the results of the realization of the AES algorithm, there are 8 files where these files have their respective functions. After the AES algorithm library is created, then the library is placed in the Arduino IDE library location so that it can be read and used to perform the encryption process on the client.

4.4 Realization of Web Server

Figure 4.7 is the realization of the webserver display with URL 127.0.0.1:5000. URL 127.0.0.1 is obtained by typing the "ifconfig" command in the terminal, while to get "5000" obtained by the "npm run dev" command in the terminal, the "npm run dev" command is also used to turn on nodemon. The webserver displays the results of the decryption of data sourced from the database. The display design on this webserver is built in visual studio code.



Thesis IIB
Darmajaya

With Universiti
Malaysia Persil

Jarak : [Safe] 113 Cm

Copyright thesis Nisa Hayati 2021. All Rights Reserved.
Version B.E.T.A

Figure 4.7 Web server View

4.5 System Testing

After the design and implementation is done, the next step is testing the design that has been made. At this stage the results of the design that have been made will be tested such as response testing, encryption and decryption testing. This test aims to determine whether the data is encrypted.

4.5.1 Functionality Testing

In functionality testing, an analysis of the suitability of the functions of the implementation results made on the client and server is carried out by design. The role of the client is to encrypt data and transmit data in the form of "distance" data obtained from the HC-SR04 sensor in the form of cipher text to the server. The server is in charge of receiving requests and resources (cipher text) from the client. The following are the results of functionality testing.

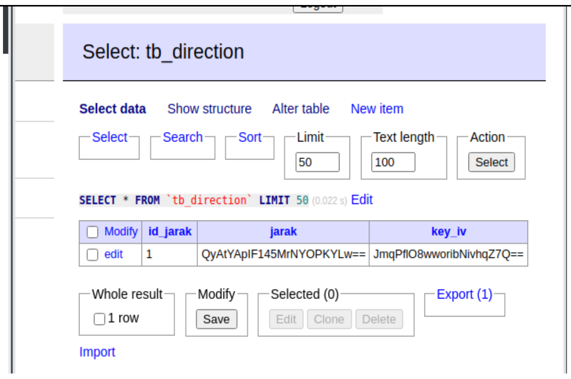
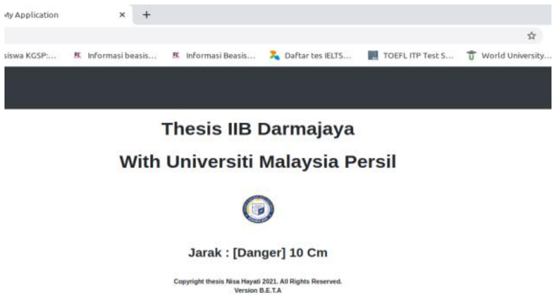
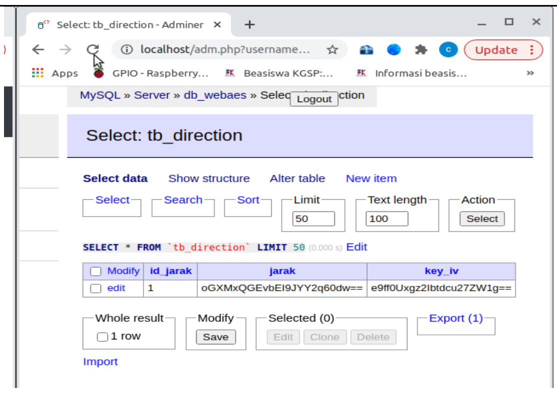
Table 4.2 Implementation Result Function

No	Test	Results
1.	Connect to a wifi network.	Succeed
2.	Reads data from the HC-SR04 sensor.	Succeed
3.	Encrypts data with 128 bit AES algorithm.	Succeed
4.	Sends HTTP requests and cipher text to the server.	Succeed
5.	Accepts HTTP requests and cipher text from clients.	Succeed
6.	Decrypts resources using the 128 bit AES algorithm.	Succeed
7.	Showing the results of the decrypt on the web server	Succeed

4.5.2 Overall Test Results

Overall testing is carried out to check overall whether the 128 bit AES algorithm can perform encryption and decryption on internet of things devices.

Table 4.3 Overall Test Results

No	Distance	Encryption and Decryption	Results
1.	10 cm	 	Succeed
2.	281 cm		Succeed

Select: tb_direction

Select data Show structure Alter table New item

Select Search Sort Limit 50 Text length 100 Action Select

```
SELECT * FROM `tb_direction` LIMIT 50 (0.022 s) Edit
```

Modify	id_jarak	jarak	key_iv
edit	1	QyAtYApIF145MrNYOPKYLw==	JmqPflO8wworibNivhqZ7Q==

Whole result 1 row Modify Save Selected (0) Edit Clone Delete Export (1)

Import

Thesis IIB Darmajaya

With Universiti Malaysia Persil

Jarak : [Danger] 10 Cm

Copyright thesis Nisa Hayati 2021. All Rights Reserved.
Version B.E.T.A

Select: tb_direction - Adminer

localhost/adm.php?username...

MySQL » Server » db_webaes » Select: tb_direction

Select: tb_direction

Select data Show structure Alter table New item

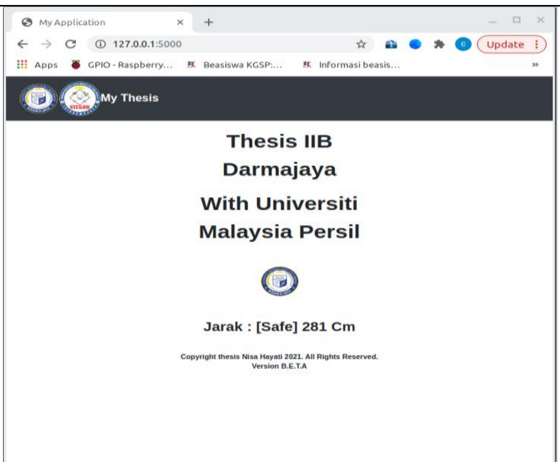
Select Search Sort Limit 50 Text length 100 Action Select

```
SELECT * FROM `tb_direction` LIMIT 50 (0.000 s) Edit
```

Modify	id_jarak	jarak	key_iv
edit	1	oGXMxQGEvbEI9JYY2q60dw==	e9ff0Uxgz2Ibtdcu27ZW1g==

Whole result 1 row Modify Save Selected (0) Edit Clone Delete Export (1)

Import

		 My Application 127.0.0.1:5000 Apps GPIO - Raspberry... Beasiswa KGSP:... Informasi beasis... My Thesis Thesis IIB Darmajaya With Universiti Malaysia Persil Jarak : [Safe] 281 Cm Copyright thesis Nisa Hayati 2021. All Rights Reserved. Version B.E.T.A	

Based on the results of research that has been done in the overall test, it can be seen that encryption and decryption of sensor data on the 128 bit AES algorithm can be done. In this overall test, it can also be seen that "distance 10 cm" will display the word "Danger" while "distance > 10" will display the word "Safe".

4.6 System Work Analysis

4.6.1 System Advantages

1. Securing internet of things device data using the AES 128 algorithm.
2. Encryption and decryption of data can run as designed.
3. Adding "general a renmdom IV" to the code in order to get a different encryption even though it has the same data. This of course adds security because the cipher text is always changing.

4.6.2 System Disadvantages

1) Only focuses on encrypting and decrypting the data obtained from the HC-SR04 sensor.
2) In this study, we do not know how long the encryption and decryption process takes.
3) This research also does not include how much memory is used during the encryption and decryption process.