Volume 9, Issue 7, July – 2024 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

Implementation of Quick Response Code Tags for Livestock Identification and Farm Management Based on Web and Android for Ruminant Farms

[1]Sindung Hadwi Widi Sasono; [2]Helmy Iskandar, Sri Kusumastuti
Polythecnic State of Semarang, Indonesia

Abstract:- Technological advancements impact various sectors of life, including animal husbandry, which is an important part of the education curriculum. Innovations are needed to optimize livestock management, especially ruminant farming, which is crucial for producing meat and milk. SMKN 1 Bawen, with its study program focusing on ruminant farming (cows, sheep, and goats), still uses conventional recording methods, leading to inefficiencies and limited information for potential buyers. To address this, a QR Code-based livestock management and identification system was developed and integrated into a website and Android application using agile methods. Black box testing confirmed that all features function well. Stress testing on the website with 2 Mbps bandwidth showed 98% successful requests and 2% failed requests, with a data transmission speed of 52,816 bits per second and a reception speed of 268,624 bits per second. At 50 Mbps bandwidth, the system had a 100% success rate, with a data sending speed of 44,304 bits per second and a receiving speed of 343,272 bits per second, demonstrating good and stable performance across different bandwidths. The Android application, tested on a POCO X3 with 6 GB RAM and 128 GB internal storage, showed a highest CPU usage of 39%, memory usage of 326 MB, and network speeds of 10.3 KB/s for data received and 535.9 KB/s for data sent. These results indicate that the application runs optimally and is suitable for use.

Keyword : Web, Android, QR Code, Livestock Management, Ruminant Farming

I. INTRODUCTION

Technological developments have greatly affected the existence of various sectors of life such as economics, politics, cultural arts, agriculture, farming, and more. In the face of scientific and technological developments in the field of farming, innovation needs to be done to improve and optimize the management of both ruminants and poultry farms. Ruminantia livestock are one of the cattle species that can provide considerable benefits in meeting human nutritional needs such as producing animal proteins through meat and milk production, therefore requiring good management in the farm enterprise. [1][2][3][8].

Farming activities are an important part of the educational curriculum, as in SMKN 1 Bawen. The focus of farming in this school involves ruminants and poultry. In the farm major in SMKN 1 Bawen specifically the ruminanti farm emphasizes the maintenance of cattle, sheep, and goats. Although students receive practical knowledge in farm management, there is a need to improve management effectiveness, especially in the context of ruminantian livestock maintenance. One of the problems with ruminantian farms there is that they still apply conventional methods in the recording of livestock information. This affects the inefficiency of the storage of recording files of cattle and also the minimum information about cattle obtained by prospective buyers [4][5].

The application of farm management manually cannot be done continuously with the rapid development of technology. Based on the problem, there is one solution to solve the problem with the implementation of Quick Response Code (QR Code) as an identification of cattle ID and web-based and Android farm management. QR Code can provide quick and accurate access. With this solution, it is expected that the management of ruminantian farms in SMK N 1 Bawen can be improved, and have a positive impact on operational efficiency and livestock welfare, while contributing to technological developments in the farm context [6][7].

II. RESEARCH METHODS

The research method used in the preparation of the final task "Implementation of Quick Response Code Tag as Animal Identification and Web-based Farm Management on Ruminantian Farms" is the agile research method. According to the book "Software Engineering: A Practitioner's Approach" [2], agile methods are one type of method in software development. It's commonly referred to as the Software Development Life Cycle. (SDLC).

Agile Software Development is a software development methodology based on a recurring process, where agreed rules and solutions are implemented in collaboration between each team in an organized and structured manner. The most important value of this agile development is enabling a team to make quick decisions, good quality and predictability, and have good potential in dealing with any change. Figure 1 is an overview of the method on this final task.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

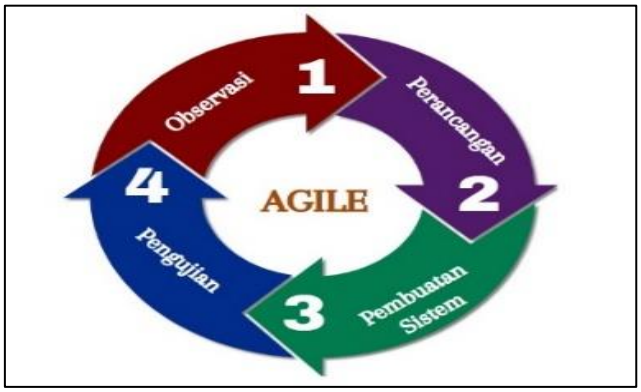

Fig 1: Agile Methods

A. Observation

The first process is observation, which involves setting goals, identifying needs, and planning steps to them. The observation begins with a survey at SMKN 1 Bawen to find out the problems faced by managers/teachers and students in farm management. The results of this survey form the background of the system building and help identify the following hardware and software needs:

- *Hardware:*

- Laptop / PC
- Android smartphones (minimum version 7.0 Nougat)

- *Software:*

- Visual Studio Code
- XAMPP
- Android Studio IDE
- Web Browser 3.2 System Design

B. System Planning

Planning is the process of designing structures to a goal that has been set during planning. The focus is on creating effective, efficient, and easilyined designs to implement system features and requirements. The farm management system architecture and QR Code can be seen in Figure 2.

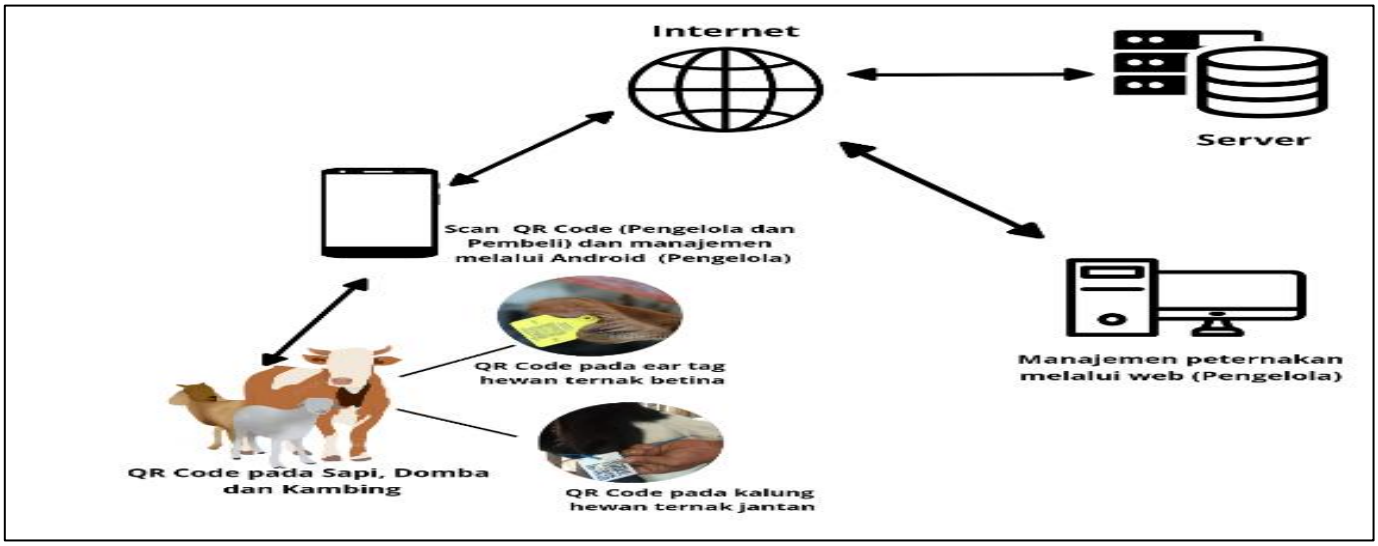

Fig 2: Farm Management System Architecture

The data entered by the admin is sent to the database over the internet. This data is then generated into a QR Code that can be accessed through the web and android. QR Codes are printed and paired to livestock using ear tags for females and neck tags for males. Farmers and buyers can scan the QR Code through the Android app.

C. Database Planning

The use case diagram represents the interaction between the system and the user. This diagram has the following structure: the left part for Buyer, the right part for Admin, and the middle part for the activities that can be performed by both. Figure 3 shows usage case diagrams for web-based and android farm management systems.

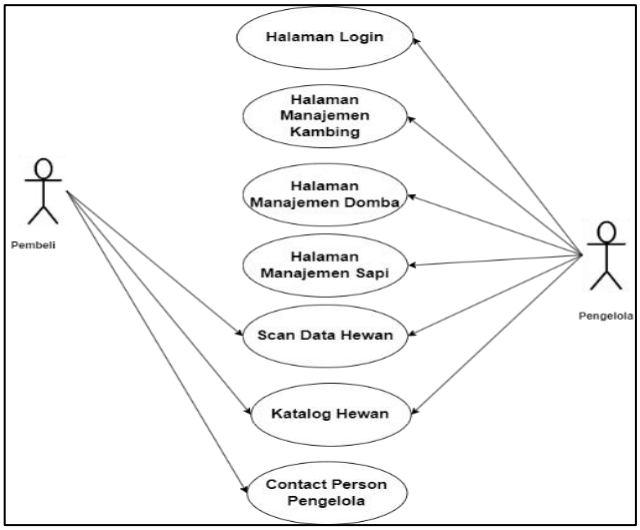

Fig 3: Usecase Diagram

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

D. Planning ERD (Entity Relationship Diagram) Database

Entity Relationship Diagram (ERD) is a technique used in the basic stages of database creation, based on an entity-relation model in which it represents how entities relate to each other in a database. (Afiifh dkk., 2022). On the QR Code system as the identification of cattle ID and management of ruminantian farms based on the web and android consists of several entities namely admin, data_domba, data_sapi, data_kambing, riwayat_berat_domba, riwayat_berat_kam, riwayat_berat_sap. The ERD view can be seen in Figure 4.

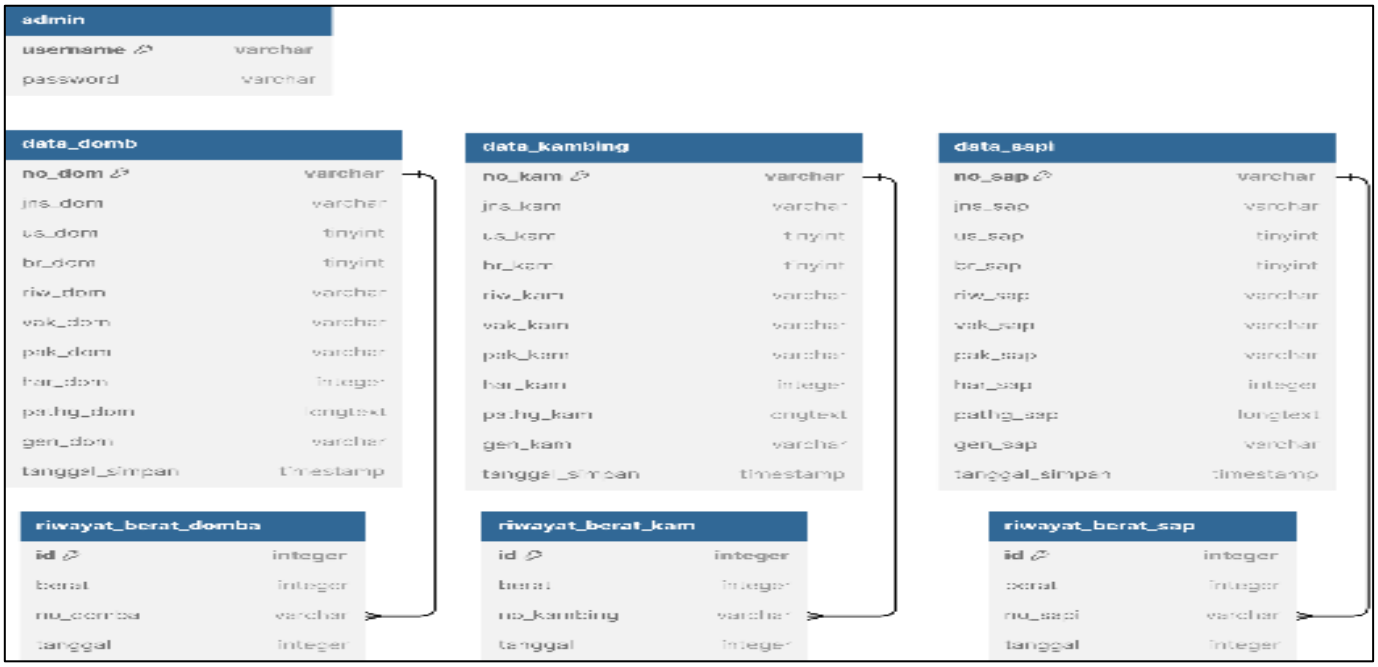

Fig 4: Entity Relationship Diagram

Database design aims to analyze the field – fields needed in building the design of this ruminantian livestock management application.

E. Test Plan

The test phase ensures that the system operates properly, is secure, and meets quality standards. Testing includes:

- Black Box Testing: Testing application functions without knowing the internal structure or code of the program, including testing the entire application system and website.
- System Performance Testing: Detects problems by running large requests, such as access by multiple users at the same time. Testing websites using Gatling Enterprise, and Android apps using Android Profiler in Android Studio.

III. SYSTEM TESTING RESULT

A. Android Interface Implementation Results

The MOPERA application home page is the first page that users see when they open the application. On this page there are login buttons and three additional menus namely the qr code scan, the animal directory, and the administrator's contact information. Implementation of the homepage can be seen in Figure 5.


Fig 5: Android Home Interface Implementation

The QR Code Scan page is a functional page for scanning the QR code that will later be installed on the cattle, the result of this QR Code scan is a detail of the animal that will be displayed using the webview that is below the scan. Implementation of the QR Code Scan page can be seen in Figure 6.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

Fig 6: Android Scan Interface Implementation

This home animal catalogue page displays three catalogue menus: goat, sheep, and cow catalogue. The aim is to enable prospective buyers to see livestock data without visiting the farm. The implementation of the animal catalogue page on the homepage can be seen in Figure 7.

The implementation of the animal catalogue page on the homepage can be seen in Figure 7.

Fig 7: Catalogue Interface Implementation

The cattle catalogue page displays the cattle data that has been added, accompanied by a detail button to get more information about each goat animal data in the catalogue. Implementation of cattle directory pages can be seen in Figure 8.

Fig 8: Implementation of the Livestock Catalogue Interface

This detail page provides complete information about the goats sold, including type, gender, age, weight, feed, price, and availability status. Designed for prospective buyers, this page also includes manager contact buttons to facilitate the purchase process. The implementation of the goat detail page in the home catalog can be seen in Figure 9.

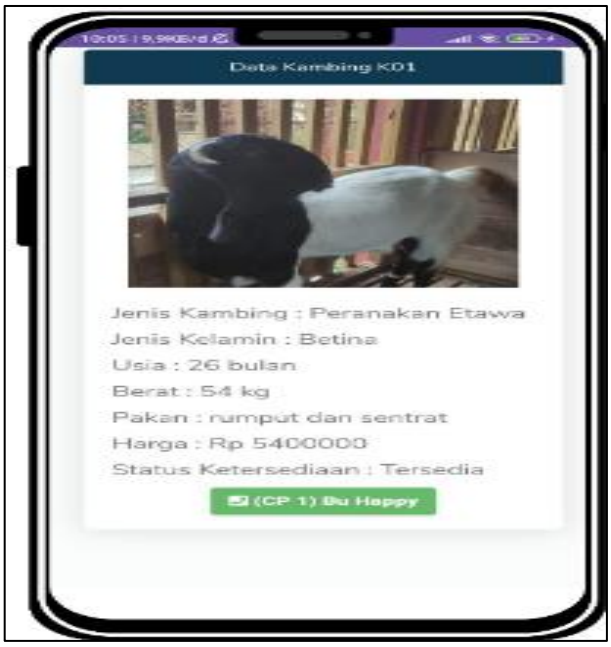

Fig 9: Animal Detail Interface Implementation

The login page makes it easy to identify the user to log into the next menu. The interface view consists of a form to enter the username and password as well as the login button for further access. The implementation of the login page can be seen in Figure 10.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

Fig 10: Login Interface Implementation

The main menu page is the page that is displayed for the first time after a user has successfully logged in. On this page there are two menus: management and monitoring. The implementation of the main menu page can be seen in figure 11.

Fig 11: Implementation of the Main Page Website Interface

After selecting the management menu, the user will be directed to the management dashboard. On the dashboard page, there are three information cards that show the number of sheep, goats, and cows. Each card has a directory button that displays the animal data that has been added, as well as a button to print the QR Code of all animal data. Implementation of the main menu page can be seen in Figure 12.

Fig 12: Android Dashoard Interface Implementation

On the animal add page there is a form used to enter the data. The data you entered will then be stored in the database and appear on the catalog page and generated to the QR Code. The goat add page view can be seen in Figure 13.

Fig 13: Implementation of an Interface Adding Livestock Data

The cattle directory page on the dashboard displays the cattle data that has been entered, accompanied by a detail button for more information about each cattle information in the catalogue, a change button to edit the data that was entered and a QR Code button to download the QR Code that contains the animal data. The implementation of the sheep catalogue page can be seen in figure 14.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

Fig 14: Implementation of the Livestock Catalogue Interface

This livestock details page displays a view of the data of cattle that has been entered into the database, where the data is displayed as no cattle, animal type, animal age, animal sex, animal weight, disease history, feeding, animal vaccine, cattle price, animal availability and animal image. The cattle data detail page also contains a graph of the weight history of the cattle located under the animal data, so that managers can maximize feeding/nutrition management. A picture of the sheep's data detail view page is shown in Figure 15.

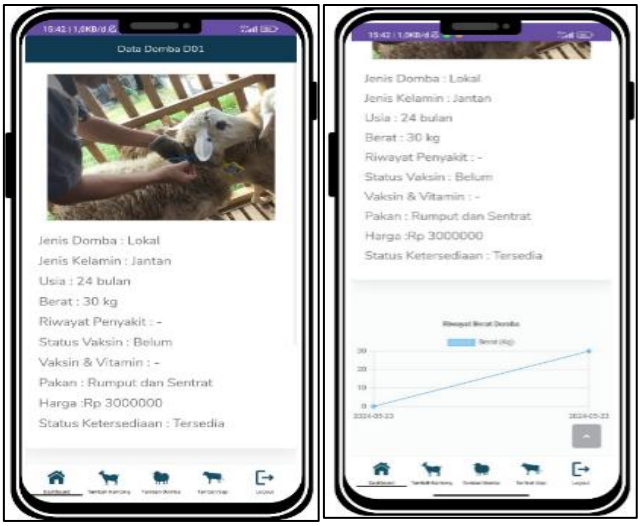

Fig 15: Implementation of a Livestock Detail Interface

The data change page on livestock serves to change and update information that has been added previously. Data that can be changed among them, animal type, sex of animal, animal age, animal history, vaccine status, type of vaccine and vitamins, animal feed, animal price, animal picture, availability status, and deleted menu. Implementation of the sheep change data page can be seen in figure 16.

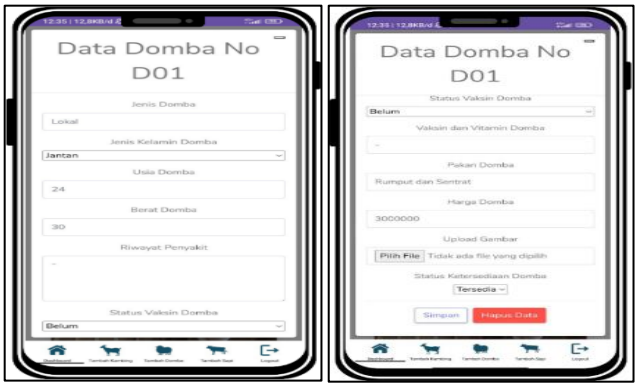

Fig 16: Animal Data Modification Interface Implementation.

Website Interface Implementation Results

The login page was first accessed by the user for identification. The interface includes a form for entering the username and password, as well as a login button for accessing the next page. The implementation of the login page interface can be seen in Figure 17.

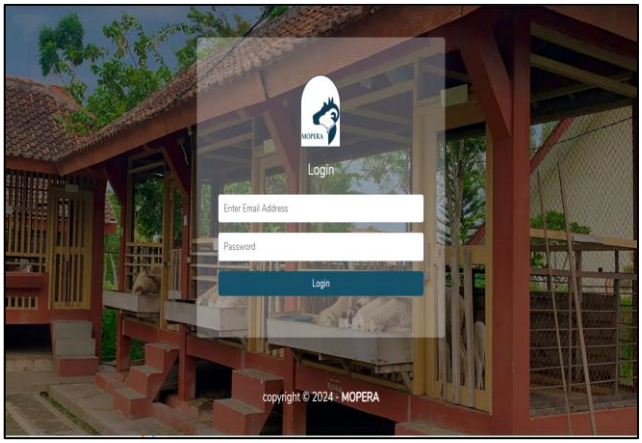

Fig 17: Implementation of Website Login Interface

Once successfully logged in, the user will be directed to the main menu page, which contains options for accessing farm management menus and monitoring menus. Figure 18 shows the implementation of the main menu page on the website.

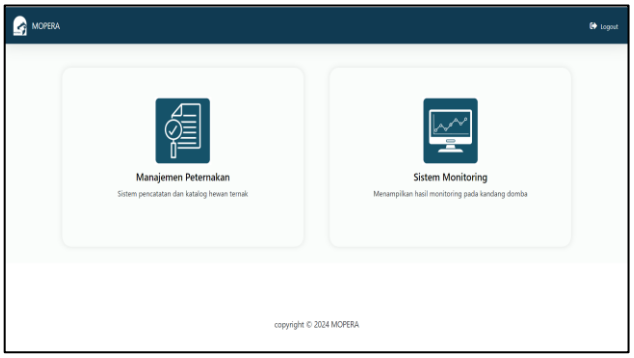

Fig 18: Implementation of the Main Page Interface of the Website

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

The dashboard page is a page after the admin successfully logged in. This page is the main page and there are some features loaded in this system that will be displayed on this page among others, menus to add animal data, display animal catalogues, and display the number of animals entered. The implementation of the dashboard page can be seen in figure 19.

Fig 19: Implementation of the Website Dashboard Interface.

The livestock catalogue page in Figure 20 displays the data of livestocks that have been entered, accompanied by a detail button for further information about each animal data.

Fig 20: Implementation of the Livestock Catalogue Interface

On the animal data add page there is a form used to add animal data. This will then be stored in the database and appear on the directory page. A page view of the animal information add page can be seen in Figure 21.

Fig 21: Data Adding Interface Implementation

This detail page provides complete information about livestock. The information displayed includes the type, gender, age, weight, disease history, vaccine status, vaccines and vitamins, feeding, prices, availability status, and weight graphs of livestocks. Implementation of the cattle data detail page on can be seen in Figure 22.

Fig. 22. Detailed implementation of livestock data

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

On the animal directory page there is a change menu that serves to change the data of the livestock that has been entered, the data that can be changed among them, type, gender, age, disease history, vaccine status, type of vaccine and vitamin, feed, price, picture, availability status, and delete menu.

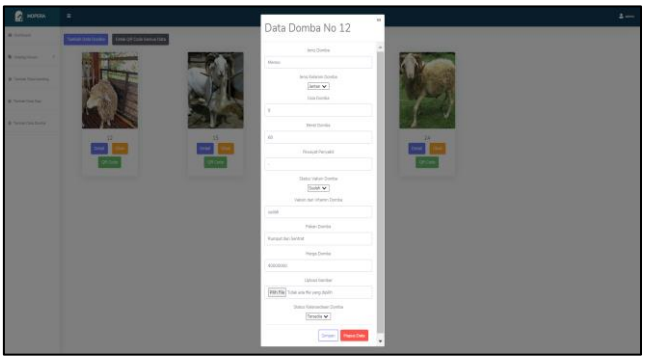

Fig 23: Implementation of an Interface to Modify Livestock Data

B. Black Box Test Results

Black box testing is a test that is performed to observe the input and output of software without knowing the code structure of the software. This test is done at the end of software development to see if the software can work properly. The aim of this test is to produce optimal software.

C. Android Performance Test Results

Testing Android application performance using Android Profiler in Android Studio monitors CPU, memory, and network usage in real time. Testing is done on HP Poco X3 NFC with 8GB of RAM and 128GB of storage.

While the application is running, there is a 39% increase in CPU usage during the QR Code scanning activity. However, at other stages, the average CPU use is below 20%, including during the login activity in which there is an authentication process, only reaches 19%. This shows that the use of the CPU does not burden the device. Figure 24 shows a graph of CPU usage at the time of Android application performance testing.

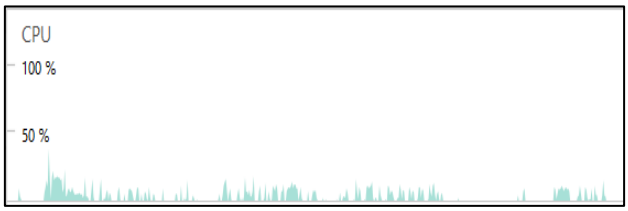

Fig 24: CPU Usage on Android Testing

When an application is running, there is an increase in memory usage when running WebView, especially when entering the Farm Management WebVIEW after logging in, which is 326 MB. However, at other stages, the average memory use remains at a stable level, that is, about 230 MB included when opening a directory which at this activity the application is indirectly connected to the database only reaches 229.5 MB.

Although there is a rise in memory use when running a WebViewer management, the application generally shows good ability in managing memory use. Thus, memory use is relatively stable and remains within the acceptable limits for such an application. Figure 25 shows a graph of memory usage on Android.

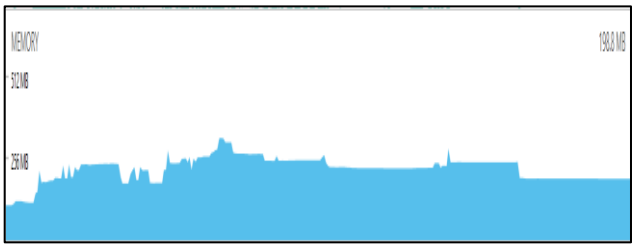

Fig 25: Memory usage on Android

During the process of adding data through form filling and uploading to the server, the application shows network usage at the highest speed of 10.3 KB/s for data received (received) and 535.9 KB/ s for data sent (sent).

The data transmitted at a rate of 535,9KB/s includes the information sent to a server, i.e. a form containing information about cattle data including images in it. During login, the data received was 83.7 KB, while the data sent was 8 KB, and the received data at the rate of 83,7 KB/S included information received from the server such as authentication responses, user information, and session tokens. The network usage on the android can be seen in figure 26 and the internet speed can be shown in figure 27.

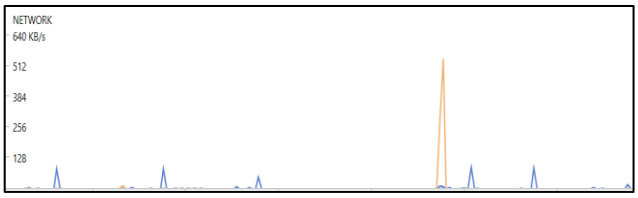

Fig 26: Network usage on Android

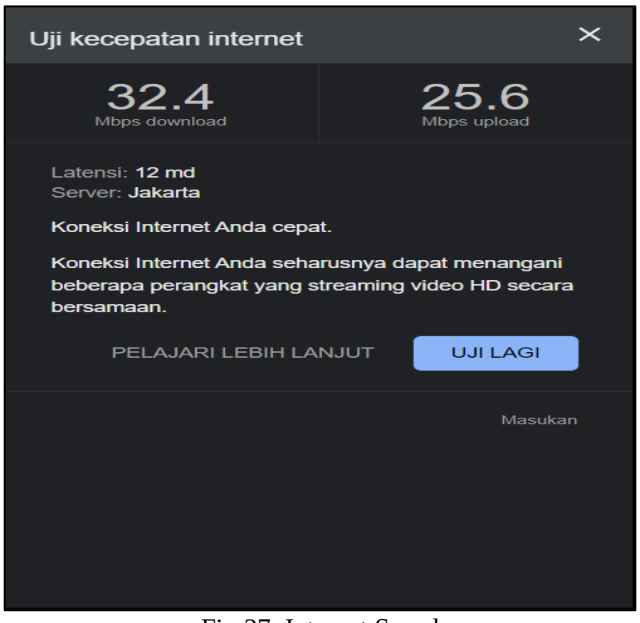

Fig 27: Internet Speed

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

D. Stress Test on Website

Stress Testing uses Gatling Enterprise, an open source tool to test the load and performance of web applications. Stress Test is used to evaluate the performance of applications under high load. In general, these tests yield several parameters, including the error ratio during the test running, total requests during the testing, and application response time.

The tests were conducted using a WiFi connection with two different bandwidths, with the aim of testing the system's response when exposed to different internet speeds. In tests with a low bandwidth of 2Mbps, a download speed of 1.75 Mbps and a upload rate of 1.81 Mbps are shown in Figure 28. In testing with a high band width of 50 Mbps, the download speed is 42.18 Mbps with a upload speed of 27.85 Mbps.

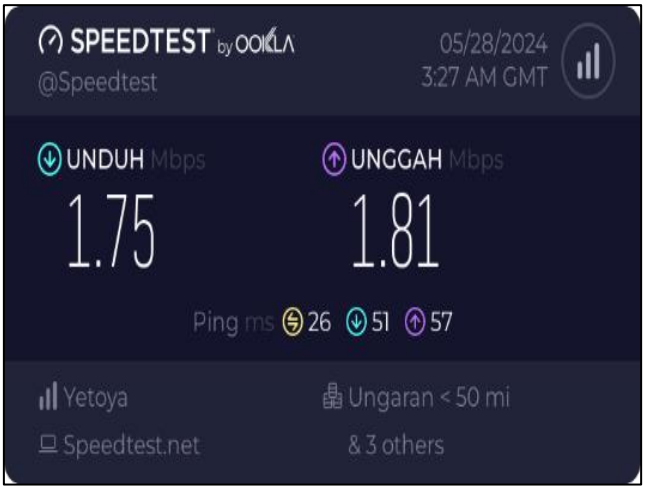

Fig 28: Low Internet Speed

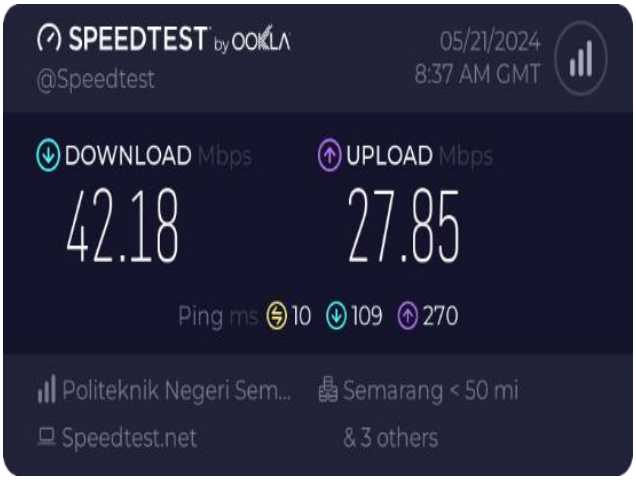

Fig 29: High Internet Speed

In a stress test with low bandwidth, the result was that the ratio of error obtained from the number of errors compared to the total number in the form of percentages, i.e. 2%. The total number of requests made during the process was 100, with the maximum number of virtual users simultaneously 15 virtual users with a response time of 4.671 seconds. The overall test results on low bandwidth are shown in Figure 30.

Fig 30: Total Test Results on Low Bandwidth

Figure 31 shows the relationship between requests and responses in seconds on tests with low bandwidth. The “OK” response indicates that the request has been successful, while the “KO” refers to the failure of the request. Requests represented by a green line indicate successful requests, with a maximum of 7 responses per second. The red line indicates a failed request that is 1 response in one second and occurs on 2 requests. Out of a total of 100 virtual user requests, 98 virtual users received an “OK” response and 2 virtual users obtained a “KO” response.

Failure to receive a response at low bandwidth is shown in the figure 32 Error per Second, which explains that the failure is due to the time it takes to perform the SSL (Secure Socket Layer) handshake process exceeding the specified time limit of 10 seconds. SSL handshaking occurs when the client and server communicate to establish a secure connection. It involves the exchange of digital certificates, identity verification, and the assignment of encryption keys.

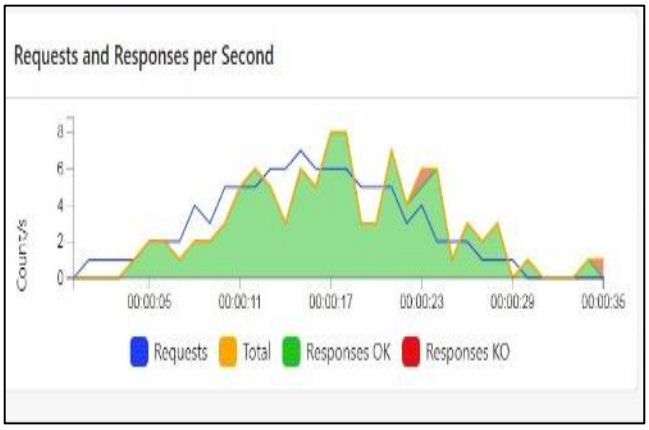

Fig 31: Request and Response Results in Second Units on Low Bandwidth

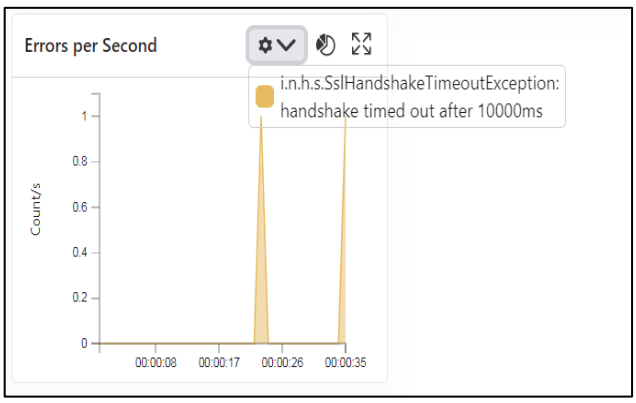

Fig 32: Error Per Second on Low Bandwidth

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

The result of CPU usage in testing with low bandwidth is shown in figure 33, "User" shows how much CPU is used to run program code. Whereas "Sys" indicates how many CPUs are used to execute system core code, such as network management. When a user interacts with an application, the usage of CPU in the user is 27% and in the sys is 4% so that the total CPU use is 31%.

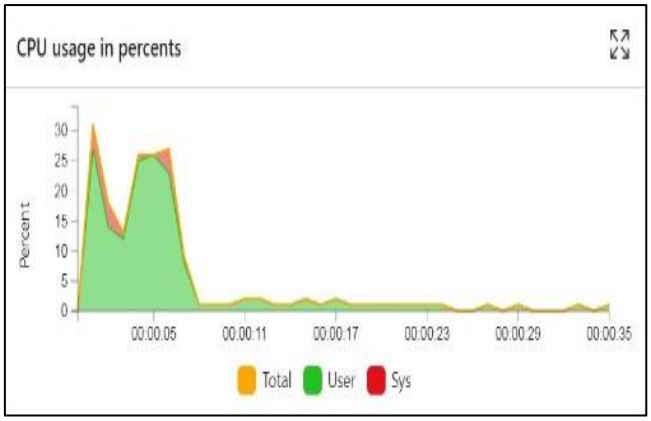

Fig 33: CPU Usage on Testing with Low Bandwidth

The use of connections in bits per second in tests with low bandwidth is shown in figure 34, which captures the measurement of data transfer speed used to measure data traffic in a stress test. In the graph, the blue line indicates the transmission of data, at speeds of 52,816 bits per second. The red line shows the reception of data at a speed of 268,624 bits a second. From this information it can be seen that the data reception speed is higher than the data transmission speed, this is due to the need to adjust the transmission rate to match the reception rate.

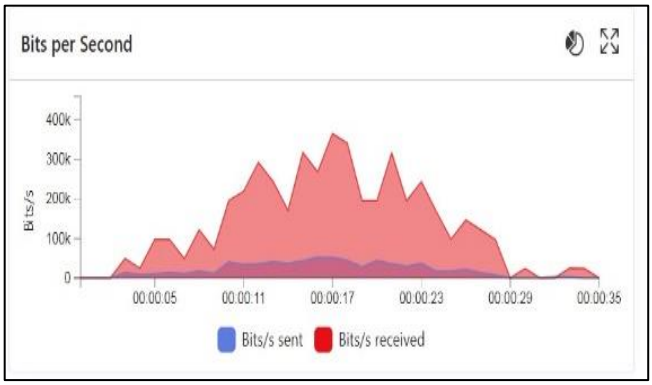

Fig 34: Use of Connections on Tests with Low Bandwidth

In the stress test with high internet speed, the result was that the ratio of error obtained from the number of errors compared to the total number in the form of percentages, i.e. 0%.

The total number of requests made during the process is 100 requests, with the maximum number of virtual users simultaneously 3 virtual users with the response time of the request 0.359s. The overall test results on high bandwidth are shown in Figure 35.

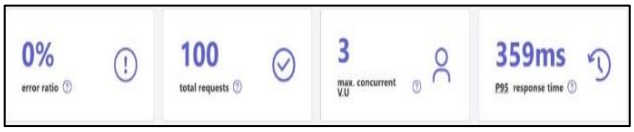

Fig 35: Overall Test Results with high bandwidth

Figure 36 shows the relationship between demand and response in seconds on tests with high bandwidth. Responses marked with "OK" indicate that the request was successful, whereas when the request failed, it is indicated with "KO". Requests represented with a green line indicate a successful request, with a maximum of 7 responses per second. The red line indicates a failed request that is 0 responses in one second.

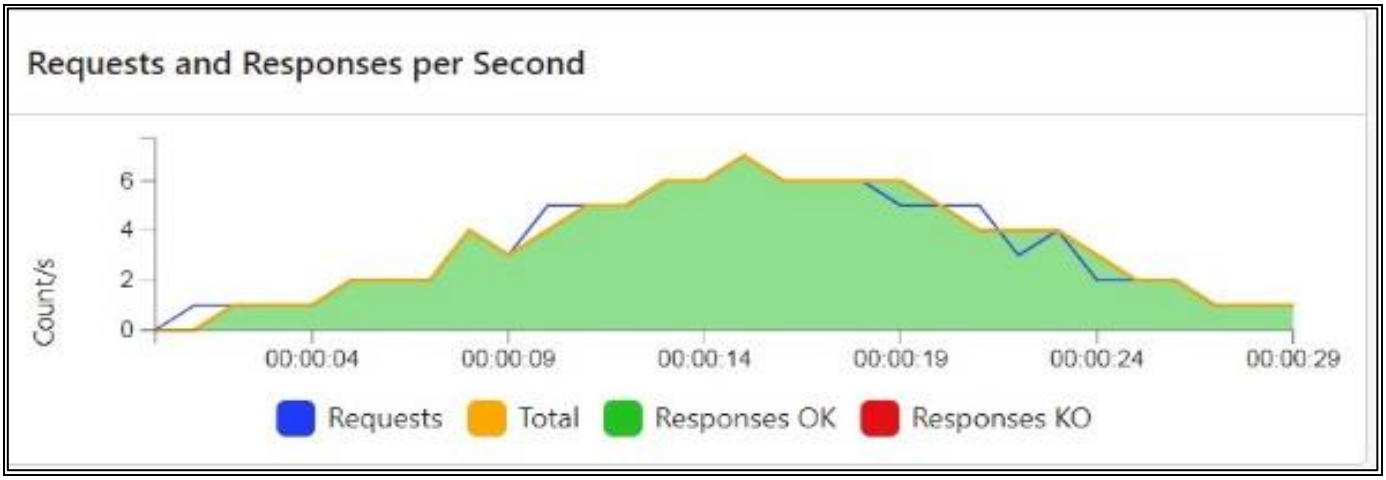

Fig 36: Request and Response Results in Second Units

The results of the CPU usage test on high bandwidth tests are shown in Figure 37. "User" shows how much CPU is used to run normal program code, like what a user does. While "Sys" indicates how many CPUs are used to execute system core code, such as network management. When users interact with applications, the CPU is used at 17%.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

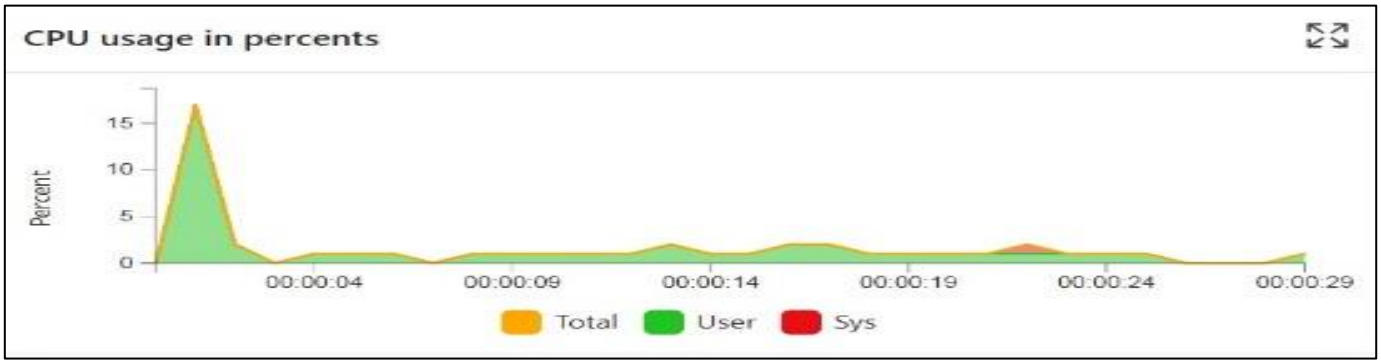

Fig 37: Results of CPU Usage on Testing with High Bandwidth

A graph of connection usage in bits per second on tests with high bandwidth is shown in figure 38. The blue line indicates the transmission of data, at speeds of 44,304 bits per second. The red line shows the reception of data at a speed of 343,272 bits a second.

It can be seen that the data reception speed is much higher than the data transmission speed, it indicates that there is an indication of a bottleneck or point in the system that limits the overall system performance, on the side of the data delivery that needs to be improved.

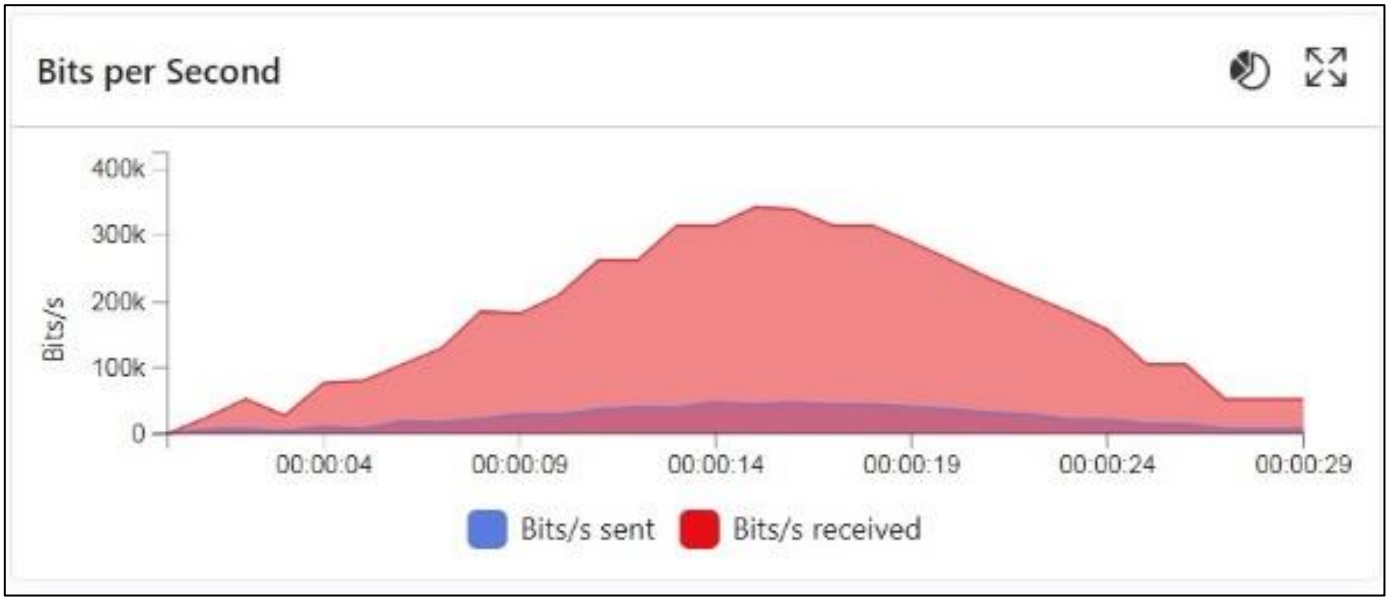

Fig 38: Use of Connections on Tests with High Bandwidth

IV. CONCLUSION

Based on the results of the planning and testing that has been carried out, conclusions can be drawn on the farm management system based on the website and android.

- Implementation of Quick Response Code Tag as identification of livestock id successfully displayed data as well as management of web-based and android farms on Romanian farms is running well.
- Testing a website using the black box testing method shows that the website interface works according to design.
- The stress test results of the website showed, with a total of 100 virtual user requests, on a 2 Mbps bandwidth the system had a success rate of 98% with 2% failure, caused by a delay in the SSL handshake process with a data reception rate of 268.624 bits per second, with an response time of 4.671s.
- On a band width of 50 Mbps with total 100 virtual users requests the system showed 100% error-free, data receipt rate of 343.272 bits/second, with response times of 359ms.
- Despite the difference in Bandwidth, the system still showed good and stable performance, with high demand success rate in both test conditions.
- Test results of Android applications showed the highest CPU usage of 39%, maximum memory consumption of 326 MB, and network usage with a highest speed of 10.3 KB/s for data received (received) and 535.9 KB / s for data sent (sent). This shows that the application has optimum performance and is worthy of use

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24JUL1889

REFERENCES

[1]. C. Trilaksana, E. Akbartama, A. Muttaqin, and O. Setyawati, "Internet of Things-based Cow Body Weight Recording System," J. EECCIS (Electrics, Electron. Commun. Control. Informatics, Syst., vol. 17, no. 1, pp. 8–12, 2023, doi: 10.21776/jeeccis.v17i1.1632.

[2]. J. G. Rajendran, M. Alagarsamy, V. Seva, P. M. Dinesh, B. Rajangam, and K. Suriyan, "IoT based tracking cattle healthmonitoring system using wireless sensors," Bull. Electr. Eng. Informatics, vol. 12, no. 5, pp. 3086–3094, 2023, doi: 10.11591/eei.v12i5.4610.

[3]. K. Subandi, H. Hermawan, and A. S. Aryani, "Value Chain Analysis Indonesian Animal Husbandry Industry," J. Appl. Sci. Adv. Technol., vol. 2, no. 1, pp. 21–28, 2019, [Online].Available: https://jurnal.umj.ac.id/index.php/JASAT/article/view/4688

[4]. R. S. Pressman, Software Quality Engineering: A Practitioner's Approach, 7th ed., vol. 9781118592. The McGraw-Hill Companies, Inc, 2014. doi: 10.1002/9781118830208.

[5]. U. Riaz, M. Idris, M. Ahmed, F. Ali, and L. Yang, "Infrared Thermography as a Potential Non-Invasive Tool for Estrus Detection in Cattle and Buffaloes," Animals, vol. 13, no. 8, 2023, doi: 10.3390/ani13081425.

[6]. V. M. T. Aleluia, V. N. G. J. Soares, J. M. L. P. Caldeira, and P. D. Gaspar, "Livestock Monitoring Prototype Implementation and Validation," Rev. Inform. Teor. e Apl., vol. 30, no. 1, pp. 53–65, 2023, doi: 10.22456/2175-2745.127207.

[7]. W. Tang, A. Biglari, R. Ebarb, T. Pickett, S. Smallidge, and M. Ward, "A smart sensing system of water quality and intake monitoring for livestock and wild animals," Sensors, vol. 21, no. 8, pp. 7–10, 2021, doi: 10.3390/s21082885

[8]. Y. Usman, "Pemberian Pakan Serat Sisa Tanaman Pertanian (Jerami Kacang Tanah , Jerami Jagung , Pucuk Tebu) Terhadap Evolusi pH , N-NH 3 dan VFA Di," J. Agripet, vol. 13, no. 2, hal. 53–58, 2013.