BAB II

TINJAUAN PUSTAKA

2.1 Teori Terkait

Penelitian yang dilakukan oleh Erlanga Eka Prasetya et al (2024). Dalam jurnal yang berjudul "Sistem *Monitoring* dan *Smart Farming* untuk Peternakan Anak Ayam Berbasis *Internet of things*(IoT)" penerapan sistem monitoring dan smart farming berbasis Internet of Things (IoT) untuk peternakan anak ayam. Proyek ini bertujuan untuk meningkatkan produktivitas dan keberlanjutan dalam sektor peternakan dengan memberikan kemampuan kepada peternak untuk memantau kondisi kandang secara *real-time*, meliputi suhu, kelembapan, dan kualitas udara, sistem yang dirancang melibatkan sensor yang terkoneksi dengan jaringan IoT, memungkinkan otomatisasi proses pemberian pakan dan air minum, serta pengelolaan data[3].

Penelitian yang dilakukan oleh Supriyono Heru et al (2021). Dalam jurnal yang berjudul "Sistem *Monitoring* Suhu dan Gas Amonia untuk Kandang Ayam Skala Kecil" menciptakan model sistem berbasis elektronik yang menggunakan komponen komersial seperti sensor DHT-11, MQ-135, Arduino Pro Mini, dan Wemos D1 untuk memantau suhu dan kadar gas amonia secara otomatis dan *real-time*. Sistem ini dirancang untuk membantu peternak menjaga kondisi kandang agar tetap optimal bagi kesehatan dan produktivitas ayam, Sistem juga dilengkapi fitur notifikasi melalui aplikasi Telegram dan

Blynk, memungkinkan peternak menerima peringatan jika kondisi melebihi batas aman[4].

Penelitian yang dilakukan oleh David C. Runtuwene et al (2024). Dalam jurnal yang berjudul ” Sistem Kontrol Dan Pemantauan Berbasis IoT untuk Kenyamanan Ternak Unggas” Perancangan dan implementasi sistem kontrol dan pemantauan berbasis IoT (*Internet of Things*) untuk kandang ayam, dengan fokus pada menjaga kenyamanan dan kesehatan ternak unggas. Sistem ini mengatasi masalah konvensional seperti pembersihan kandang yang masih dilakukan secara manual dan tidak teratur, serta kesulitan dalam memantau suhu dan kadar amonia yang optimal. Dengan menggunakan sensor DHT22 untuk mengukur suhu dan kelembapan, serta sensor MQ135 untuk mendeteksi kadar amonia, sistem ini dapat mengatur suhu kandang antara 30°C - 34°C dan menjaga kadar amonia di bawah 25 ppm. Mikrokontroler ESP32 berfungsi sebagai pengontrol utama dan perangkat komunikasi IoT, memungkinkan pemantauan kondisi kandang secara *real-time* melalui aplikasi *smartphone*[5].

Penelitian yang dilakukan oleh Try Handoyo et al (2022). Dalam jurnal yang berujudul” Sistem Monitoring Suhu Dan Kelembaban Pada Kandang Anak Ayam Broiler Berbasis *Internet Of Things*” mengimplementasikan sistem monitoring suhu dan kelembaban pada kandang anak ayam broiler. Sistem ini dirancang untuk memantau dan mengontrol suhu serta kelembaban kandang secara *real-time,* tujuannya adalah meningkatkan efektivitas pemantauan kondisi kandang, memastikan suhu dan kelembaban tetap ideal untuk

pertumbuhan anak ayam boiler, serta memudahkan peternak dalam memantau kondisi kandang dari jarak jauh melalui antarmuka website[6].

Penelitian yang dilakukan oleh Wibowo et al (2022). Dalam jurnal yang berjudul ”Implementasi *Internal Controller of* Kandang *Close House* berbasis IoT” mengimplementasikan sistem kontrol dan monitoring berbasis *Internet of Things* (IoT) pada kandang *close house* untuk meningkatkan produktivitas ayam broiler, serta memantau kondisi kandang secara *real-time* melalui perangkat *mobile.* Tujuannya adalah meminimalkan stres pada ayam akibat perubahan lingkungan, meningkatkan efisiensi operasional, dan mengurangi ketergantungan pada tenaga manual, sehingga peternak dapat memantau dan mengontrol kandang dari jarak jauh dengan lebih efektif[7].

2.1.1 Penelitian Terkait

Tabel 2. 1 Penelitian Terkait

Penulis	Judul	Fokus	Tools
Erlanga Eka Prasetya	Sistem *Monitoring* dan *Smart Farming* untuk Peternakan Anak Ayam Berbasis *Internet of things*(IoT)	Memantau kondisi kendang meluputi, suhu, kelembapan, kualitas udara, pakan, dan minum menggunakan bly nk berbasis IoT.	Blynk

Penulis	Judul	Fokus	Tools
Supriyono Heru	Sistem *Monitoring* Suhu dan Gas Amonia untuk Kandang Ayam Skala Kecil	Pengukuran Suhu dan Kadar Gas Amonia pada Perangkat Bergerak dengan Aplikasi Telegram dan Blynk	Telegram dan Blynk
David C. Runtuwene	Sistem Kontrol Dan Pemantauan Berbasis IoT untuk Kenyamanan Ternak Unggas	Kadar amonia serta suhu kandang ayam dapat dikontrol dengan sistem otomatis dan dapat dipantau menggunakan blynk	Blynk
Try Handoyo	Sistem Monitoring Suhu Dan Kelembaban Pada Kandang Anak Ayam Broiler Berbasis *Internet Of Things*	Monitoring dan kontroling suhu, kelembaban pada kandang melalui *website*	Website

Penulis	Judul	Fokus	Tools
Wibowo	Implementasi *Internal Controller of* Kandang *Close House* berbasis IoT	Meningkatkan kualitas maupun kuantitas produksi ayam broiler pada kandang *close house*	Android Studio

2.1.2 Penelitian yang diteliti

Tabel 2. 2 Penelitian yang diteliti

Penulis	Judul	Fokus	Tools
M. Nur Julianto, Rudi Heryansyah	Brocass: Sistem Kandang Pintar Berbasis Andorid Untuk Pemeliharaan Anak Ayam Boiler	Monitoring suhu, kelembapan, minum otomatis dan kontrol pakan kandang melalui aplikasi android.	Android Studio, Firebase

2.2 Landasan Teori

2.2.1 Peternakan Anak Ayam Boiler

Para peternak ayam boiler saat ini masih banyak yang menggunakan cara tradisonal atau manual yang dilakukannya untuk merawat anak ayam boiler yang mereka punya seperti pemberian pakan, minum, kebersihan kandang yang masih manual, dan

mencangkup lingkup pengawasan pada kandang secara berkala.Hal ini dapat menghambat beberapa faktor penting, seperti pertumbuhan anak ayam dan asupan pakan yang diberikan. Anak ayam membutuhkan lingkungan dengan suhu optimal (32–35°C pada usia awal) dan kelembapan yang stabil. Dengan kondisi kandang yang optimal dan stabil, stres pada anak ayam dapat diminimalkan, sehingga pertumbuhan mereka meningkat. Peternakan ayam broiler merupakan salah satu sektor yang mempunyai peran terhadap perkembangan ekonomi Indonesia. Kualitas dan kuantitas produksi daging ayam broiler berkaitan dengan beberapa hal seperti sistem manajemen perkandangan [8].

2.2.2 Sistem

Sistem didefinisikan sebagai sekumpulan komponen yang saling terhubung dan bekerja secara terkoordinasi untuk mencapai tujuan tertentu melalui proses yang terstruktur. Komponen-komponen ini dapat mencakup perangkat keras, perangkat lunak, dan jaringan komunikasi yang dirancang untuk menjalankan fungsi tertentu secara efisien. Dalam konteks teknologi, sistem memungkinkan pengelolaan data dan otomatisasi proses untuk mendukung pengambilan keputusan yang akurat dan cepat[9].

2.2.3 Monitoring

Monitoring adalah proses mengamati, mengukur, dan mencatat secara sistematis aktivitas atau kondisi suatu sistem, proses, atau objek dalam jangka waktu tertentu untuk memastikan bahwa semuanya berjalan sesuai rencana atau untuk mendeteksi masalah sedini mungkin. Monitoring dilakukan secara berkelanjutan agar informasi yang diperoleh selalu akurat dan terkini, sehingga dapat digunakan sebagai dasar pengambilan keputusan yang tepat. Proses ini juga membantu dalam mengevaluasi efektivitas suatu program atau sistem, serta menjadi alat kontrol untuk menjaga kinerja dan kualitas tetap optimal sesuai dengan tujuan yang telah ditetapkan[10].

2.2.4 Kontroling

Kontroling adalah proses mengamati, mengevaluasi, dan mengarahkan aktivitas atau hasil kerja untuk memastikan bahwa semua kegiatan berjalan sesuai dengan rencana, standar, atau tujuan yang telah ditetapkan. Dalam proses ini, dilakukan perbandingan antara hasil aktual dengan standar atau target yang direncanakan, serta diambil tindakan korektif jika terdapat penyimpangan atau deviasi. Kontroling bertujuan untuk menjaga konsistensi, meningkatkan efisiensi, serta memastikan bahwa organisasi, sistem, atau proses dapat mencapai tujuan secara efektif. Proses ini merupakan salah satu fungsi penting dalam manajemen untuk menjamin keberhasilan pelaksanaan kegiatan[11].

2.2.5 *Internet of Things(IoT)*

Merupakan sistem yang terdiri dari berbagai perangkat yang dibekali dengan sensor, software, dan teknologi pendukung lainnya, yang memungkinkan perangkat-perangkat tersebut saling terhubung dan berinteraksi melalui jaringan internet. Tujuan dari penerapan IoT adalah untuk mengotomatisasi berbagai aktivitas, meningkatkan efisiensi operasional, serta mendukung pengambilan keputusan yang lebih cepat dan akurat berdasarkan data yang diperoleh secara langsung dan *real-time*[12].

2.2.6 Aplikasi Android

Aplikasi android adalah sebuah sistem operasi perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya, Google inc memberi android inc merupakan pendatang baru yang membuat perangkat lunak untuk ponsel atau *smartphone*[13].

2.2.7 Android Studio

Android studio adalah *Integrated Development Environment* (IDE) resmi dari google yang digunakan oleh pengembang untuk membuat aplikasi android. Android studio memungkinkan pengembang untuk membuat aplikasi dengan mudah dan cepat dengan bantuan fitur-fitur yang disediakan. Android Studio menyediakan berbagai fitur, seperti editor kode, emulator Android untuk menguji

aplikasi, alat *debugging*, dan fitur lainnya yang mendukung pengembangan aplikasi[14].

Gambar 2. 1 Logo Android Studio

2.2.8 Bahasa Java

Java merupakan bahasa pemrograman yang telah hadir sejak tahun 1995 dan dikenal karena ekosistemnya yang luas. Bahasa ini tidak hanya digunakan untuk membuat aplikasi Android, tetapi juga digunakan pada berbagai platform lainnya seperti aplikasi web, desktop, dan *server.* Salah satu kelebihannya adalah kemampuan untuk menulis kode satu kali dan menjalankannya di berbagai sistem melalui *Java Virtual Machine* (JVM). Selama bertahun-tahun, Java menjadi bahasa andalan bagi pengembang aplikasi Android karena kestabilannya dan dukungan komunitas yang kuat[15].

2.2.9 Bahasa Kotlin

Kotlin merupakan bahasa pemrograman modern terbaru yang stabil bertujuan untuk memudahkan pengembang. Kotlin memiliki keunggulan-keunggulan tertentu jika dibandingkan dengan bahasa pemrograman lain. Keunggulan dari Kotlin salah satunya ada sintaks

Kotlin lebih ringkas, *typesafe*, interoperabilitas dengan Java maupun bahasa pemrograman lain, dan memberikan banyak cara untuk dapat menggunakan kembali kode dari berbagai platform untuk menunjang produktivitas pengembangan[16].

2.2.10 Firebase

Firebase merupakan sebuah teknologi dari perusahaan raksasa Google berbentuk platform yang tujuannya untuk memudahkan pengembangan sistem yang menggunakan sebuah *resource* REST API dalam pengembangan aplikasi berbasis android atau *mobile* untuk berkomukasi dengan *server* umumnya menggunakan REST API akan tetapi dalam proses pembuatannya sangatlah lama karena beberapa faktor seperti keamanan, kecepatan, dan kemudahan akses, firebase hadir untuk memangkas kegiatan pengembangan REST API tersebut sehingga mempermudah pengembang aplikasi dalam pembuatan aplikasi [17].

Gambar 2. 2 Firebase

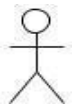
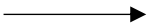
2.2.11 UML *(Unified Modeling Language)*

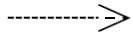
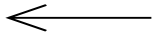
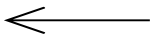
Unified Modeling Language (UML) adalah alat yang paling umum digunakan untuk membuat dokumentasi piranti perangkat lunak. Bahasa ini telah menjadi standar industri untuk visualisasi, desain, dan dokumentasi sistem piranti perangkat lunak. UML memiliki fungsi untuk membantu pendeskripsian dan desain system perangkat lunak, khususnya system yang dibangun menggunakan pemrograman berorientasi objek [18].

a. *Use case* Diagram

Diagram ini memperlihatkan himpunan *usecase* dan aktor-aktor, diagram ini sangat penting untuk mengorganisir dan terdapat aktor yang merupakan sebuah gambaran entitas dari manuasia atau sebuah sistem yang melakukan pekerjaan di sistem.

Tabel 2. 3 *Use case* Diagram

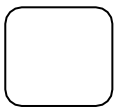
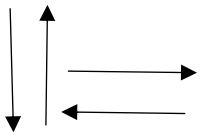
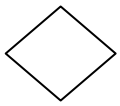
Simbol	**Keterangan**
	Aktor : Menspesifikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *Use Case*.
	*Use Case* : Abstraksi dan interaksi antara sistem dan aktor
	*Association* : Abstraksi dari interaksi penghubung antara aktor dengan *use case*

Simbol	Keterangan
	*Generalisasi* : Menunjukan spesialisasi aktor untuk dapat berpatipasi dengan *use case*
<<include>> 	Menunjukan bahwa suatu *use case* seluruhnya merupakan fungsional dari *use case* lainnya
<<extend>> 	Menunjukan bahwa suatu *use case* merupakan tambahan fungsional dari *use case* lainnya jika suatu kondisi terpenuhi

b. *Activity* Diagram

Sebuah diagram aktivitas UML menggambarkan perilaku dinamis dari suatu sistem atau bagian dari sistem melalui aliran kontrol antara aksi yang dilakukan sistem.

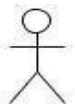
Tabel 2. 4 *Activity* Diagram

Simbol	Nama	Keterangan
	*Activity*	Memperlihatkan bagaimana masing- masing kelas antar muka saling berinteraksi satu sama lain.
	*Action*	State dari sistem yang mencerminkan eksekusi suatu aksi.
	*Initial Node*	Bagaimana objek dibentuk atau diawali.
	*Final Node*	Bagaimana objek dibentuk dan diakhiri.
	*Line Connector*	Digunakan untuk menggambarkan suatu keputusan / tindakan yang harus diambil pada kondisi tertentu
	*Decision*	Pilihan untuk mengambil keputusan

c. *Sequence* Diagram

Menggambarkan interaksi antar objek didalam dan di sekitar sistem (termasuk pengguna, display dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan output tertentu.

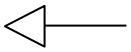
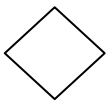
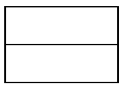
Tabel 2. 5 *Sequence* Diagram

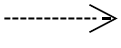
Simbol	**Nama**	**Keterangan**
	*Actor*	Menggambarkan orang yang sedang berinteraksi dengan sistem
	*Entity Class*	Menggambarkan hubungan kegiatan yang akan dilakukan.
	*Control Class*	Menggambarkan penghubung antara Boundary denga tabel
	Message	Mengindikasikan komunikasi antara objek dengan objek.
	*Life Line*	Objek *entity,* antar muka yang saling berinteraksi.

d. *Class* Diagram

Diagram *Class* bersifat statis. Dimana dalam ini memperlihatkan himpunan kelas, antarmuka, kolaborasi serta relasi.

Tabel 2. 6 *Class* Diagram

Simbol	Nama	Keterangan
	*Generalization*	Hubungan dimana objek anak (*descendent*). Berbagai perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
	*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari dua objek.
	*Class*	Himpunan dari objek-objek yang berbagai atribut serta operasi yang sama.
	*Collaboration*	Deskripsikan dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu faktor.
	*Realization*	Operasi yang benar- benar dilakukan suatu objek.

Simbol	Nama	Keterangan
	*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.

2.2.12 *Black Box* Testing

Pengujian *black box* testing disebut sebagai pengujian perilaku. Dimana struktur interior, logika perangkat lunak yang diuji tidak diketahui oleh penguji. Penguji didasarkan kepada spesifikasi kebutuhan dan tidak perlu dilakukannya analisis kode. Pengujian black box testing pengujian ini dilakukan dari sudut pandang pengguna akhir[19].

Beberapa jenis pengujian pada *black box* testing, diantaranya seperti; partisi, analisis nilai batas, grafik penyebab efek, pengujian orthogonal array, pengujian transisi negara, dan fuzzing.