



BAB II

TINJAUAN PUSTAKA

2.1. Teori Umum

2.1.1. Smartphone

Williams & Sawyer (2011), smartphone adalah telepon selular dengan mikroprosesor, memori, layar dan modem bawaan. Smartphone merupakan ponsel multimedia yang menggabungkan fungsionalitas PC dan handset sehingga menghasilkan gadget yang mewah, di mana terdapat pesan teks, kamera, pemutar musik, video, game, akses email, tv digital, search engine, pengelola informasi pribadi, fitur GPS, jasa telepon internet dan bahkan terdapat telepon yang juga berfungsi sebagai kartu kredit.

2.1.2. Pengertian Android

Safaat (2015:1), android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi.

Salbino (2010:7), android merupakan sistem operasi berbasis Linux yang bersifat terbuka (*open source*) dan dirancang untuk perangkat seluler layar sentuh seperti *smartphone* dan komputer tablet.

2.2. Teori Judul

2.2.1. Pengertian Aplikasi

Huda (2012), pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan maupun pemrosesan data yang diharapkan.

Mulyadi (2010), pomponen yang berguna melakukan pengolahan data maupun kegiatan-kegiatan seperti pembuatan dokumen atau pengolahan data. Aplikasi adalah bagian PC yang berinteraksi langsung dengan user.



2.2.2. Pengertian Extreme Programming

Paradigma pembangunan mencakup seperangkat aturan dan praktik yang terjadi dalam konteks kerangka empat kegiatan yaitu: perencanaan, desain, coding, dan pengujian. Keempat aktivitas inilah yang akan menghasilkan sebuah perangkat lunak yang didasari dengan konsep model Extreme Programming. (Pressman 2010).

2.2.3. Pengertian Quick Response Code (QR Code)

QR Code merupakan sebuah simbol penandaan objek nyata yang terbuat dari pola batang-batang berwarna hitam dan putih agar mudah untuk dikenali oleh komputer (Rahayu, Yeni Dwi, dkk. 2006).

2.2.4. Pengertian Peternakan

Peternakan adalah segala urusan yang berkaitan dengan sumber daya fisik, benih, bibit dan/atau bakalan, pakan, alat dan mesin peternakan, budi daya ternak, panen, pascapanen, pengolahan, pemasaran, dan pengusahannya (UU No. 18 Tahun 2009 tentang Peternakan dan Kesehatan Hewan Pasal 1 angka 1).

2.2.5. Pengertian Ternak

Ternak adalah hewan peliharaan yang produknya diperuntukan sebagai penghasil pangan, bahan baku industri, jasa, dan/atau hasil ikutannya yang terkait dengan pertanian (UU No. 18 Tahun 2009 tentang Peternakan dan Kesehatan Hewan Pasal 1 angka 5).

2.3. Teori Khusus

2.3.1. Usecase Diagram

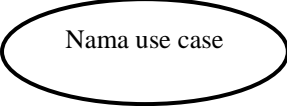
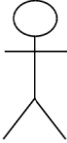
Sukanto dan Shalahuddin (2013:155), *Usecase* atau diagram *usecase* merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat. *Usecase* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat.

Syarat penamaan pada *usecase* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *usecase* yaitu pendefinisian apa yang disebut aktor dan *usecase*.

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Usecase* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol-simbol yang ada pada diagram *usecase* menurut Rosa dan Shalahuddin (2013:156):


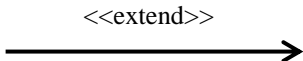
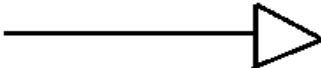
Tabel 2.1 Simbol-simbol diagram *use case*


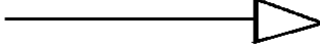
| No. | Simbol | Keterangan |
|-----|--|--|
| 1. |  <p><i>Use case</i> Nama use case</p> | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama <i>use case</i> |
| 2. |  <p>Aktor / actor nama aktor</p> | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, |



| | | |
|--|--|---|
| | | tapi aktor belum tentu merupakan orang. |
|--|--|---|

Lanjutan Tabel 2.1 Simbol-simbol *use case* diagram

| No | Simbol | Keterangan |
|----|---|---|
| 3. | Asosiasi / <i>association</i>  | Komunikasi antar aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan actor |
| 4. | Ekstensi / <i>extend</i>  | Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu; prinsip inheritance pada pemrograman berorientasi objek; ditambahkan, misal arah panah mengarah pada <i>use case</i> yang ditambahkan; biasanya <i>use case</i> yang menjadi <i>extend</i> -nya merupakan jenis yang sama dengan <i>use case</i> yang menjadi induknya. |
| 5. | Generalisasi / <i>generalization</i>  | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya |



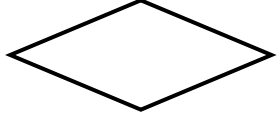

| | | |
|----|--|--|
| 6. | <p>Menggunakan / <i>include</i> / <i>uses</i></p> <p><<include>></p>  <p><<uses>></p>  | <p>Relasi use case tambahan ke sebuah <i>use case</i> di mana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat <i>use case</i>.</p> |
|----|--|--|


Sumber : Rosa dan Shalahuddin (2013:156)

2.3.2. Activity Diagram

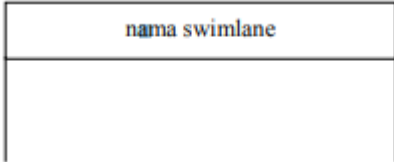
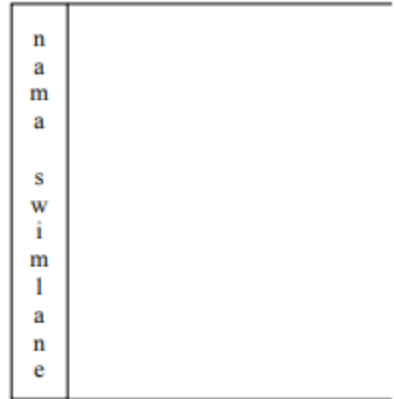
Sukanto dan Shalahuddin (2013:161), Diagram aktivitas atau activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Berikut adalah simbol-simbol yang ada pada diagram aktivitas menurut Rosa dan Shalahuddin (2013:162):

Tabel 2.2 Simbol-simbol *activity diagram*

| No. | Simbol | Keterangan |
|-----|--|---|
| 1. | <p>Status Awal</p>  | <p>Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal</p> |
| 2. | <p>Aktivitas</p>  | <p>Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja</p> |
| 3. | <p>Percabangan / <i>decision</i></p>  | <p>Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu</p> |
| 4. | <p>Penggabungan / <i>join</i></p>  | <p>Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu</p> |

| | | |
|----|---|---|
| 5. | Status Akhir  | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
|----|---|---|

Lanjutan Tabel 2.2 Simbol-simbol *activity diagram*

| No. | Simbol | Keterangan |
|-----|---|--|
| 6. | Swimlane  atau  | Swimlane memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

Sumber : Rosa A.S dan M. Shalahudin (2014:162)

2.3.3. Class Diagram

Sukamto dan Shalahuddin (2013:141), Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.

2. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas main

Kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

2. Kelas yang menangani tampilan sistem (*view*)

Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.

3. Kelas yang diambil dari pendefinisian use case (*controller*)

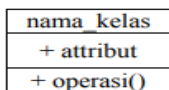
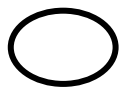

Kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian use case, kelas ini biasanya disebut dengan kelas proses yang menangani proses bisnis pada perangkat lunak.



4. Kelas yang diambil dari pendefinisian data (*model*)

Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

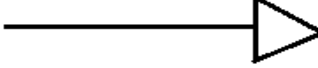

Berikut adalah simbol-simbol yang ada pada diagram kelas menurut Rosa dan Shalahuddin (2013:146):

Tabel 2.3 Simbol-simbol *class diagram*

| No. | Simbol | Keterangan |
|-----|---|--|
| 1. | Kelas  | Kelas pada struktur sistem |
| 2. | Antarmuka / <i>interface</i>  nama_ <i>interface</i> | Sama dengan konsep interface dalam pemrograman berorientasi objek |
| 3. | Asosiasi / association  | Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity |

| | | |
|----|---|--|
| 4. | Asosiasi berarah / <i>directed association</i>  | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity |
| 5. | Agregasi / <i>aggregation</i>  | Relasi antarkelas dengan makna semua- bagian (whole-part) |

Lanjutan Tabel 2.3 Simbol-simbol *class diagram*

| No. | Simbol | Keterangan |
|-----|---|--|
| 6. | Generalisasi  | Relasi antarkelas dengan makna generalisasi-spesialisasi (umum khusus) |
| 7. | Kebergantungan / <i>dependency</i>  | Relasi antarkelas dengan makna kebergantungan antarkelas |

Sumber : Rosa A.S dan M. Shalahudin (2014:146)

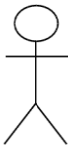
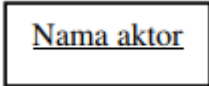

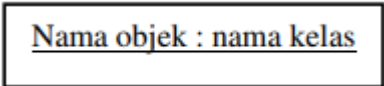
2.3.4. *Sequence Diagram*


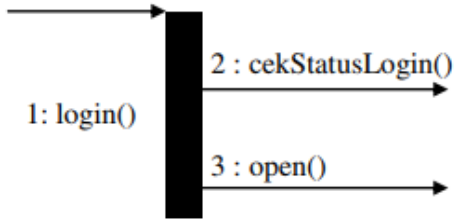
Sukanto dan Shalahuddin (2013:165), *Sequence diagram* atau diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek.

Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak use case yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.



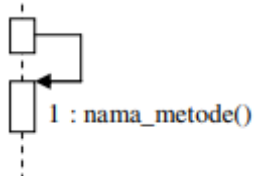
Berikut adalah simbol-simbol yang ada pada diagram sekuen menurut Rosa dan Shalahuddin (2013:146):


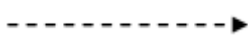
Tabel 2.4 Simbol-simbol *sequence diagram*

| No. | Simbol | Keterangan |
|-----|---|---|
| 1. | Aktor  nama aktor atau  tanpa waktu aktif | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda diawal frase nama aktor. |
| 2. | Garis Hidup / lifeline  | Menyatakan kehidupan suatu objek |
| 3. | Objek  | Menyatakan objek yang berinteraksi pesan |

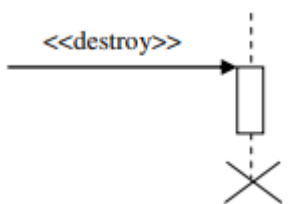
| | | |
|----|--|--|
| 4. | <p>Waktu Aktif</p>  | <p>Menyatakan objek dalam keadaan aktif dan berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya</p>  <p>Maka cekStatusLogin() dan open()</p> |
|----|--|--|

Lanjutan Tabel 2.4 Simbol-simbol *sequence diagram*

| No. | Simbol | Keterangan |
|-----|---|--|
| 5. | <p>Pesan tipe <i>create</i></p> <p style="text-align: center;"><<create>></p>  | <p>Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat</p> |
| 6. | <p>Pesan tipe <i>call</i></p> <p style="text-align: center;">1 : nama_metode()</p>  | <p>Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,</p>  <p>Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi</p> |

| | | |
|----|---|--|
| 7. | Pesan tipe <i>send</i> 1 : masukkan  | Menyatakan bahwa suatu objek mengirimkan data/masukkan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim |
| 8. | Pesan tipe <i>return</i> 1 : keluaran  | Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian |

Lanjutan Tabel 2.4 Simbol-simbol *sequence diagram*

| No. | Simbol | Keterangan |
|-----|--|--|
| 9. | Pesan tipe <i>destroy</i>  | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaliknya jika ada create maka ada destroy |

Sumber : Rosa A.S dan M. Shalahudin (2014:146)

2.4. Teori Program

2.4.1. Pengertian Android

Android merupakan system operasi untuk perangkat bergerak berbasis Linux yang mencakup system operasi, middleware, dan aplikasi (Safaat, 2012). Menurut Safaat, Android memiliki kelebihan antara lain:

a. Lengkap (*Complete platform*)

Android menyediakan banyak tools dalam membangun perangkat lunak dan memungkinkan untuk peluang mengembangkan aplikasi.



b. Terbuka (*Open source platform*)

Android disediakan melalui linsensi open source dimana pengembang dapat dengan bebas mengembangkan aplikasi.

c. Free (*Free platform*)

Android merupakan platform/aplikasi yang bebas dimana tidak ada linsensi atau biaya royalty untuk dikembangkan pada Android. Aplikasi untuk Android juga dapat didistribusikan dan diperdagangkan dalam bentuk apapun.

2.4.2. Pengertian Java

Sukanto dan Shalahuddin (2013:103) menjelaskan, Java adalah bahasa pemrograman objek murni karena semua kode programnya dibungkus dalam kelas.

Jubilee (2015:1), Java adalah bahasa pemrograman berbasis objek yang bersifat *Write Once, Run Anywhere* (Program yang ditulis satu kali dan dapat berjalan pada banyak platform).

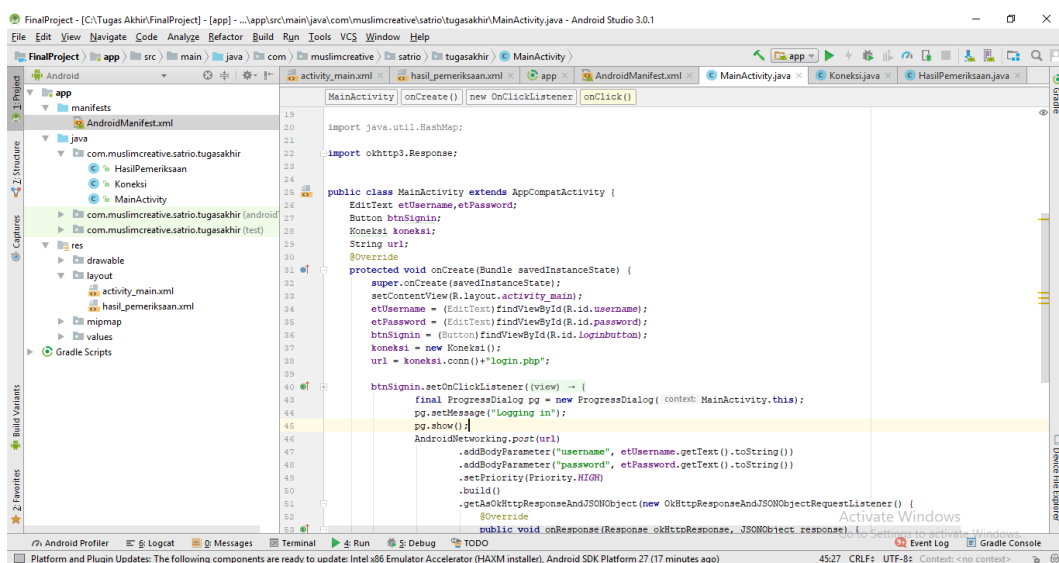
Ramadhani (2015:5), Java adalah bahasa pemrograman yang dapat dijalankan diberbagai computer termasuk ponsel. Java merupakan bahasa pemrograman bersifat umum/nonspesifik (*general purpose*), dan secara khusus didesain untuk memanfaatkan dependensi implementasi seminimal mungkin.

2.4.3. Pengertian Firebase

Firebase memiliki produk utama, yaitu menyediakan database realtime dan backend sebagai layanan (Backend as a Service). Layanan ini menyediakan pengembang aplikasi API yang memungkinkan aplikasi data yang akan disinkronisasi di klien dan disimpan di cloud Firebase ini. Firebase menyediakan library untuk berbagai client platform yang memungkinkan integrasi dengan Android, iOS, JavaScript, Java, Objective-C dan Node aplikasi Js dan dapat juga disebut sebagai layanan DbaaS (Database as a Service) dengan konsep realtime. 8 Firebase digunakan untuk mempermudah dalam penambahan fitur-fitur yang akan dibangun oleh developer.

2.4.4. Pengertian *Android Studio*

Android Studio adalah sebuah *Integrated Development Environment (IDE)* untuk *platform* Android. *Android Studio* tersedia bagi pengembang untuk mencoba secara gratis. *Android Studio* berada di awal tahap *preview access* mulai dari versi 0.1 pada Mei 2013, kemudian memasuki tahap beta mulai dari versi 0.8 yang dirilis pada bulan Juni 2014. Berdasarkan software JetBrains 'IntelliJ IDEA,



Android Studio dirancang khusus untuk pengembangan Android.

(http://en.wikipedia.org/wiki/Android_Studio)

Gambar 2.1 Tampilan *Android Studio*

2.5. Penelitian Terkait

Berdasarkan penelitian yang telah dilakukan oleh Faraj, Adian Fatchur Rochim dan Rinta Kridalukmana (2015), menjelaskan tentang Aplikasi mobile pengawasan ternak yang dapat mengolah data ternak dengan risiko data hilang atau rusak lebih kecil dengan metode *Ionic*. Penelitian ini dibuat untuk mencari alternatif untuk pengolahan data dari konvensional menjadi digital.

Alim Rahmat Rido, Jusak dan Julianto Lemantara (2014) melakukan penelitian yang bertujuan untuk dapat memberikan konsultasi *online* untuk



mengidentifikasi jenis penyakit pada hewan sapi dan memberikan saran pengobatan dini secara tepat dan menghasilkan beberapa laporan untuk digunakan oleh kepala Dinas Peternakan Kabupaten Blitar.

Bambang Sugiantoro dan Fuad Hasan (2015), melakukan pembuatan Aplikasi QR Code Scanner yang dibangun berbasis Android menggunakan ZBar Library mempunyai fitur Scanner, Bookmark, tentang Museum Sonobudoyo, dan menampilkan foto koleksi secara 3D Rotate. Semua fitur dalam aplikasi ini dapat berjalan dengan baik.

Berdasarkan penelitian yang telah dilakukan oleh Faisal Rahman¹, M. Husni dan Wahyu Hidayat (2016), menjelaskan tentang aplikasi yang dapat digunakan untuk memberikan penjadwalan jurnal dan SOP harian yang harus dilakukan oleh pegawai sebagai reminder untuk pegawai dalam melaksanakan tugas hariannya didalam kandang. Aplikasi ini mampu menampilkan data stok gudang baik berupa pakan, obat maupun vaksin agar pegawai dapat menggunakan stok yang ada dengan baik, agar dikemudian hari kehabisan stok pada gudang bisa diantisipasi dengan tepat.

Andriano Eucharistia Wibowo, Herman Tolle dan Ratih Kartika Dewi (2015), melakukan pembuatan aplikasi berbasis web yang berfungsi untuk memformulasikan pakan ternak sapi dari dua jenis bahan, yaitu hijauan dan konsentrat untuk memenuhi nutrisi harian atau pun meningkatkan berat badan sapi. Aplikasi bernama SunRan (Susun Ransum) tersebut menggunakan framework Yii dalam pengembangannya.