LAPORAN AKHIR
PENGEMBANGAN SISTEM BERORIENTASI OBJEK
DEPARTEMEN ILMU KOMPUTER IPB
SEMESTER GENAP 2018/2019

Nama Aplikasi : CowPlan
Nama Developer : Kelompok 14
Anggota Developer : Hanifah Izzaty - G64160001
: Aar Riana - G64160003
: Wafa Ainina A - G64160014

Deskripsi Singkat Aplikasi

CowPlan adalah sebuah aplikasi berbasis *Mobile App* untuk penanggalan masa birahi sapi. Aplikasi ini berisi tampilan kalender dimana peternak dapat menandai tanggal melahirkan sapi. Kemudian secara otomatis pada kalender akan ditandai tanggal birahi yang merupakan hasil *current date* ditambah 21. Menandai tanggal birahi sapi ini perlu dilakukan karena beberapa peternak seringkali lupa dengan jadwal birahi sapi yang mereka miliki, sehingga harus menunggu periode selanjutnya untuk dibirahi. Hal ini tentunya akan mempengaruhi produktifitas peternakan itu sendiri. Dengan aplikasi ini diharapkan para peternak tidak lagi lupa dan melewatkan jadwal birahi sapi yang mereka miliki, sehingga dapat meningkatkan produktifitas.

User analysis

Berisi tentang:

- analisis user

  User dari aplikasi ini adalah seorang peternak yang tentunya merupakan seorang laki-laki/perempuan dewasa. Seorang peternak ada yang memiliki pendidikan tinggi maupun tidak.
- user story
  - Sebagai user saya bisa menambahkan tanggal sapi melahirkan sehingga dapat mengetahui prediksi tanggal birahi sapi.
  - Sebagai user saya bisa melihat seluruh data sapi yang pernah dimasukkan sehingga dapat mengetahui siklus melahirkan sapi.

Spesifikasi teknis lingkungan pengembangan

Perangkat Keras:

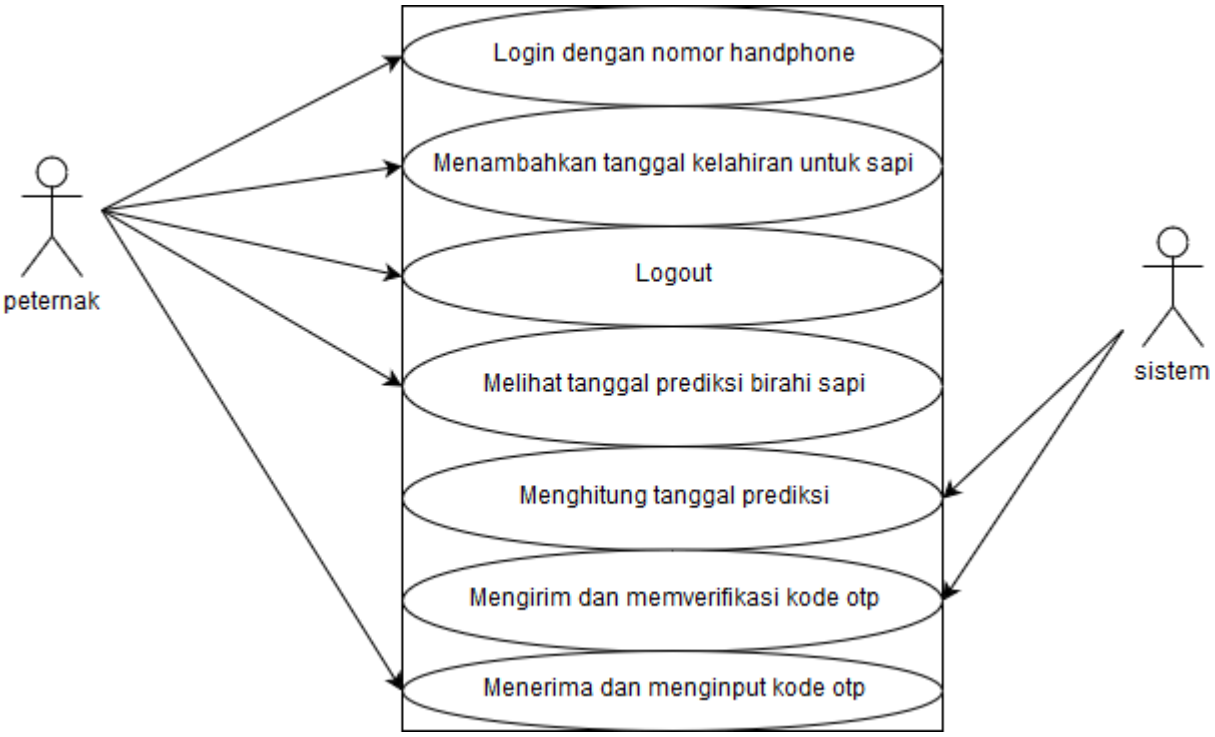
- Processor : Intel Core i5 7200U 3.1GHz
- RAM : 4GB
- VGA : NVIDIA GeForce 930MX

Perangkat Lunak:

- Android Studio
- Sistem Operasi Windows 10
- Google Firebase

Hasil dan pembahasan

Use Case :



Terdapat dua aktor utama yang berperan dalam aplikasi, yaitu peternak dan sistem. Peternak dapat masuk ke aplikasi dengan memasukkan nomor handphonenya, kemudian sistem akan mengirimkan kode otp ke nomor tersebut. Task utama dari aplikasi ini adalah memprediksi tanggal birahi sapi berdasarkan tanggal terakhir sapi tersebut melahirkan. Peternak akan memasukkan tanggal melahirkan seekor sapi, dan sistem yang akan langsung menghitung tanggal birahi sapi dan menampilkannya di halaman utama.

Use case description

Usecase Name : menandai tanggal sapi melahirkan	ID: UC01	Importance Level: High
Primary Actor: Peternak		Use Case Type: detail, essential
Stakeholder and Interest : Peternak – ingin menandai tanggal sapi melahirkan		
Brief Description : di dalam use case ini dijelaskan bagaimana peternak memberi penanggalan untuk sapi ketika melahirkan		
Trigger : peternak ingin memaksimalkan produksi dari sapi melalui siklus kehamilannya		

Type : temporal
Relationship : Association : peternak Include : - Extend : - Generalization : memprediksi berdasarkan tanggal sebelumnya
Normal Flow of Events : 1. Peternak menandai tanggal sapi melahirkan
Sub Flows : -
Alternate/Exceptional Flows : Prediksi tidak sesuai

Usecase Name : melihat seluruh data sapi yang pernah dimasukkan	ID: UC02	Importance Level: High
Primary Actor: Peternak		Use Case Type: detail, essential
Stakeholder and Interest : Peternak – ingin melihat seluruh data sapi yang pernah dimasukkan		
Brief Description : di dalam use case ini dijelaskan bagaimana peternak melihat seluruh data sapi yang pernah dimasukkan		
Trigger : peternak ingin mengetahui siklus melahirkan sapi. Type : temporal		
Relationship : Association : peternak Include : - Extend : - Generalization : -		
Normal Flow of Events : 1. Peternak melihat seluruh data sapi yang pernah dimasukkan		
Sub Flows : -		
Alternate/Exceptional Flows : Terdapat data yang tidak ditampilkan		

Hasil implementasi program

Pembuatan model untuk data sapi

```java
public class data_sapi {

    //Deklarasi Variable
    private String nama_sapi;
    private String tanggal_birahi;
    private String tanggal_melahirkan;
    private String key;

    public String getKey() { return key; }

    public void setKey(String key) { this.key = key; }

    public String getNama_sapi() { return nama_sapi; }

    public void setNama_sapi(String nama_sapi) { this.nama_sapi = nama_sapi; }

    public String getTanggal_birahi() { return tanggal_birahi; }

    public void setTanggal_birahi(String tanggal_birahi) { this.tanggal_birahi = tanggal_birahi; }

    public String getTanggal_melahirkan() { return tanggal_melahirkan; }

    public void setTanggal_melahirkan(String tanggal_melahirkan) {
        this.tanggal_melahirkan = tanggal_melahirkan;
    }
```

Login User

```java
private void signInWithPhoneAuthCredential(PhoneAuthCredential credential) {
    mAuth.signInWithCredential(credential)
            .addOnCompleteListener( activity: this, (task) → {
                    if (task.isSuccessful()) {
                        // Sign in success, update UI with the signed-in user's information
                        Log.d(TAG, msg: "signInWithCredential:success");

                        FirebaseUser user = task.getResult().getUser();

                        //check if user exist or not in Database

                        userID = user.getUid();

                        myRef.child("users").child(userID).addListenerForSingleValueEvent(new ValueEventListener() {
                            @Override
                            public void onDataChange(DataSnapshot snapshot) {
                                if (snapshot.getValue() != null) {

                                    Intent y = new Intent( packageContext: PhoneAuthActivity.this, MainActivity.class);
                                    y.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                                    y.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
                                    startActivity(y);
```

Input data sapi (create)

```java
private void submitSapi(data_sapi sapi) {
    /**
     * Ini adalah kode yang digunakan untuk mengirimkan data ke Firebase Realtime Database
     * dan juga kita set onSuccessListener yang berisi kode yang akan dijalankan
     * ketika data berhasil ditambahkan
     */
    FirebaseUser currentFirebaseUser = FirebaseAuth.getInstance().getCurrentUser();
    String userid = currentFirebaseUser.getUid();

    database.child("sapi").child(userid).push().setValue(sapi).addOnSuccessListener( activity: this, (OnSuccessListener) (aVoid) → {
            nama_sapi.setText("");
            //Snackbar.make(findViewById(R.id.button), "Data berhasil ditambahkan", Snackbar.LENGTH_LONG).show();
    });
}
```

Melihat data sapi (read)

```
public void onDataChange(DataSnapshot dataSnapshot) {

    /**
     * Saat ada data baru, masukkan datanya ke ArrayList
     */
    daftarSapi = new ArrayList<>();
    for (DataSnapshot noteDataSnapshot : dataSnapshot.getChildren()) {
        /**
         * Mapping data pada DataSnapshot ke dalam object Sapi
         * Dan juga menyimpan primary key pada object Sapi
         * untuk keperluan Edit dan Delete data
         */
        data_sapi sapi = noteDataSnapshot.getValue(data_sapi.class);
        sapi.setKey(noteDataSnapshot.getKey());

        /**
         * Menambahkan object Sapi yang sudah dimapping
         * ke dalam ArrayList
         */
        daftarSapi.add(sapi);
    }
    /**
     * Inisialisasi adapter dan data sapi dalam bentuk ArrayList
     * dan mengeset Adapter ke dalam RecyclerView
     */
    adapter = new AdapterSapiRecycleView(daftarSapi,  ctx: BacaSapiActivity.this);
    rvView.setAdapter(adapter);
}
```

Fungsi prediksi tanggal birahi sapi

```
public void hitung_period(){
    Calendar c = Calendar.getInstance();
    int mYear = c.get(Calendar.YEAR);
    int mMonth = c.get(Calendar.MONTH);
    int mDay = c.get(Calendar.DAY_OF_MONTH);

    // display the current date
    String CurrentDate = mYear + "/" + mMonth + "/" + mDay;

    String dateInString = CurrentDate; // Start date
    SimpleDateFormat sdf = new SimpleDateFormat( pattern: "yyyy-MM-dd");

    c = Calendar.getInstance();

    try {
        c.setTime(sdf.parse(dateInString));
    } catch (ParseException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    c.add(Calendar.DATE,  i1: 21);//insert the number of days you want to be added to the current date
    sdf = new SimpleDateFormat( pattern: "dd/MM/yyyy");
    Date resultdate = new Date(c.getTimeInMillis());
    dateInString = sdf.format(resultdate);
    theDate.setText(dateInString);
}
```

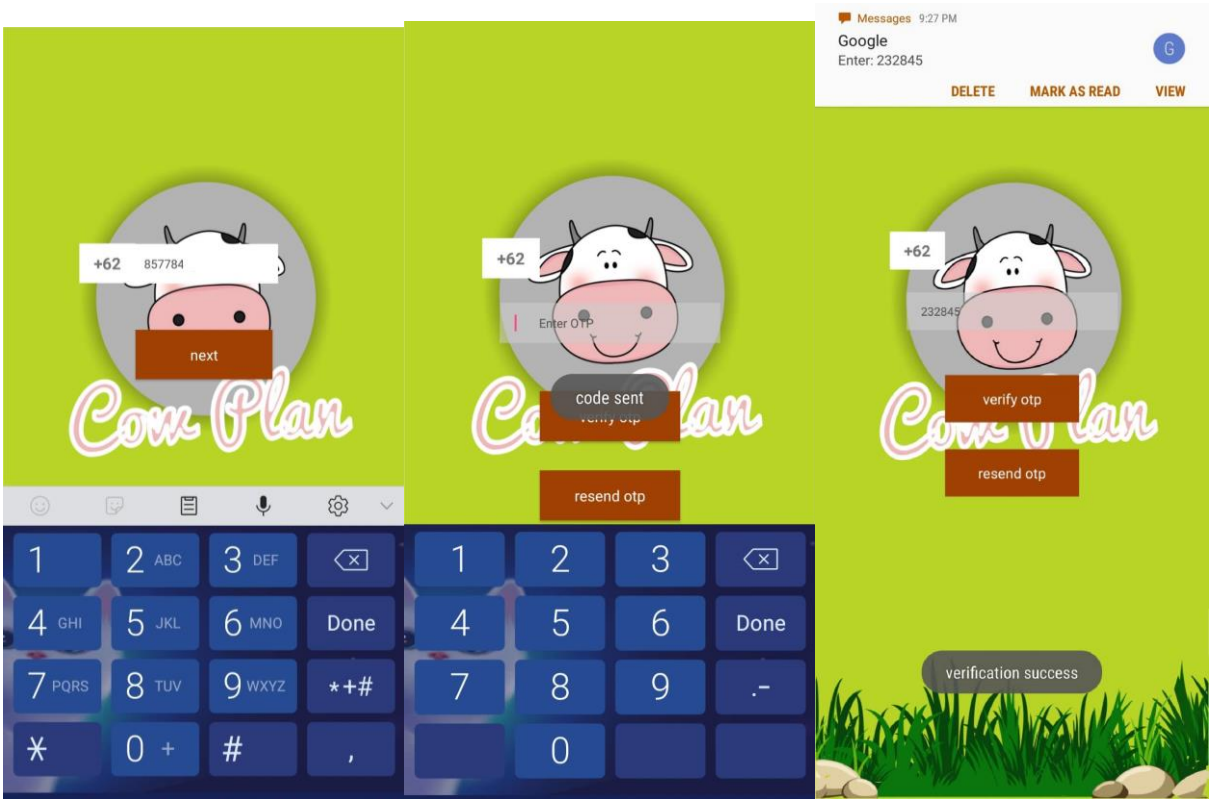
Menghapus Data

```
@Override
public void onDeleteData(data_sapi sapi, final int position) {
    /**
     * Kode ini akan dipanggil ketika method onDeleteData
     * dipanggil dari adapter lewat interface.
     * Yang kemudian akan mendelete data di Firebase Realtime DB
     * berdasarkan key barang.
     * Jika sukses akan memunculkan Toast
     */
    if(database!=null){database.child("sapi").child(sapi.getKey()).removeValue().addOnSuccessListener(new OnSuccessListener<Void>() {
        @Override
        public void onSuccess(Void aVoid) {
            Toast.makeText(BacaSapiActivity.this,"success delete", Toast.LENGTH_LONG).show();
        }
    });
```

Proses Pengujian aplikasi
Metode pengujian sistem yang diterapkan disini adalah metode *black box testing*.

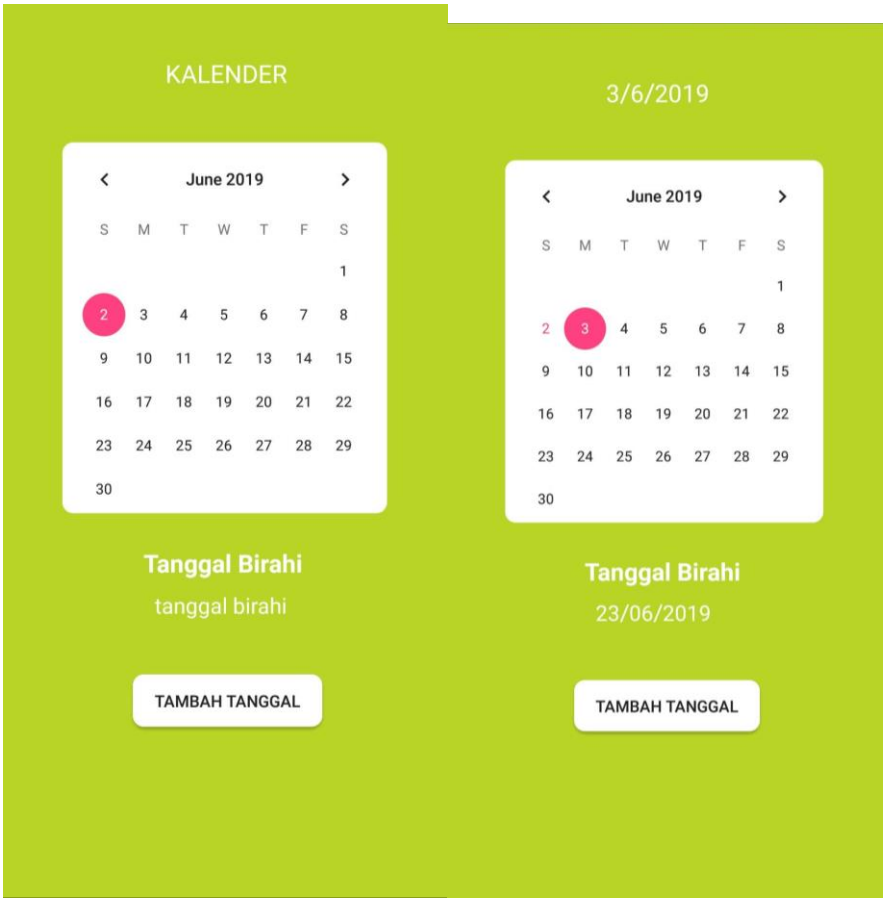
Proses Login

Mengisi No HP Menerima OTP Berhasil Login

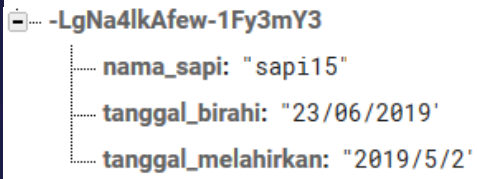


Prediksi Sapi

Sebelum tanggal dipilih Setelah tanggal dipilih

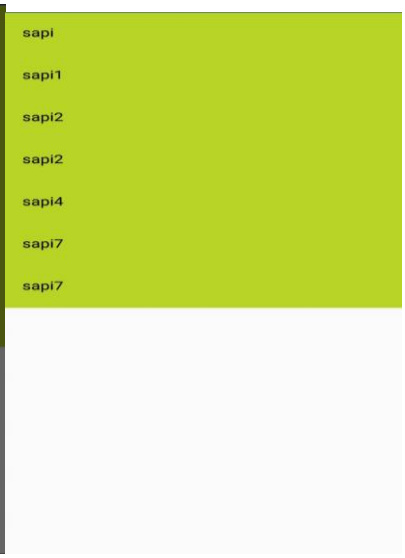


Menginput Data



Melihat Data sapi

Menghapus Data



Saran

Untuk pengembangan selanjutnya diharapkan dapat menambahkan fitur notifikasi untuk memudahkan peternak sebagai penyempurna tujuan dari aplikasi ini.