REPUBLIK INDONESIA
KEMENTERIAN HUKUM DAN HAK ASASI MANUSIA

SURAT PENCATATAN
CIPTAAN

Dalam rangka pelindungan ciptaan di bidang ilmu pengetahuan, seni dan sastra berdasarkan Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta, dengan ini menerangkan:

Nomor dan tanggal permohonan	:	EC00202135464, 26 Juli 2021
Pencipta		
Nama	:	**Yasser A. Djawad, S.T., M.Sc., Ph.D., Dr. Hendra Jaya, S.Pd., M.T. dkk**
Alamat	:	Jln. A.P. Pettarani, Makassar, SULAWESI SELATAN, 90222
Kewarganegaraan	:	Indonesia
Pemegang Hak Cipta		
Nama	:	**Universitas Negeri Makassar**
Alamat	:	Jln. A.P. Pettarani, Makassar, SULAWESI SELATAN, 90222
Kewarganegaraan	:	Indonesia
Jenis Ciptaan	:	**Program Komputer**
Judul Ciptaan	:	**Aplikasi Tombol Darurat Covid-19 Berbasis IoT**
Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia	:	1 Agustus 2020, di Makassar
Jangka waktu pelindungan	:	Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.
Nomor pencatatan	:	000262595

adalah benar berdasarkan keterangan yang diberikan oleh Pemohon.
Surat Pencatatan Hak Cipta atau produk Hak terkait ini sesuai dengan Pasal 72 Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

a.n. MENTERI HUKUM DAN HAK ASASI MANUSIA
DIREKTUR JENDERAL KEKAYAAN INTELEKTUAL

Dr. Freddy Harris, S.H., LL.M., ACCS.
NIP. 196611181994031001

LAMPIRAN PENCIPTA

No	Nama	Alamat
1	Yasser A. Djawad, S.T., M.Sc., Ph.D.	Jln. A.P. Pettarani
2	Dr. Hendra Jaya, S.Pd., M.T.	Jln. A.P. Pettarani
3	Ridwansyah, S.T., M.T.	Jln. A.P. Pettarani
4	Suhartono, S.Kom., M.Kom.	Jln. A.P. Pettarani
5	Ahmad Risal, S.Pd., M.Pd.	Jln. A.P. Pettarani
6	Sutarsi Suhaeb, S.T., M.Pd.	Jln. A.P. Pettarani
7	Nurhayyun Dahri, S.Pd.	Jln. A.P. Pettarani
8	Nasruddin, S.Pd.	Jln. A.P. Pettarani

Oleh:

Dr. Yasser A. Djawad, S.T., M.Sc

Dr. Hendra Jaya, S.Pd., M.T.

Ridwansyah, S.T., M.T.

Suhartono, S.Kom., M.Kom.

Ahmad Risal, S.Pd., M.Pd.

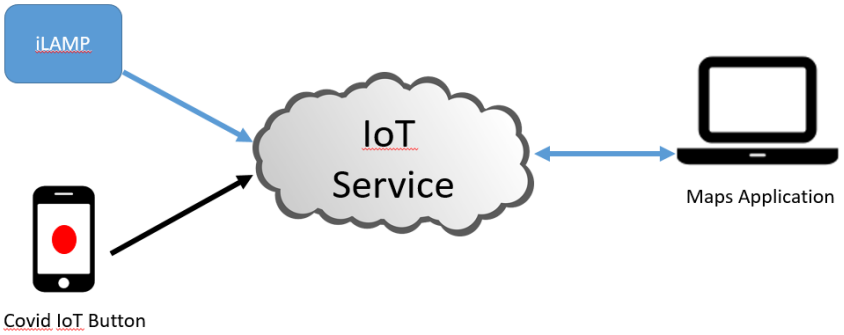
Sutarsi Suhaeb, S.T, M.Pd.

Nurhayyun Dahri, S.Pd.

Nasruddin, S.Pd.

2.1. System design

Sistem terdiri dari Aplikasi tombol darurat Covid-19 IoT, sebuah perangkat internet Loop-mediated isothermal amplification (iLAMP) dan website yang berfungsi pangkalan data utama seperti yang ditunjukkan ada gambar 1.

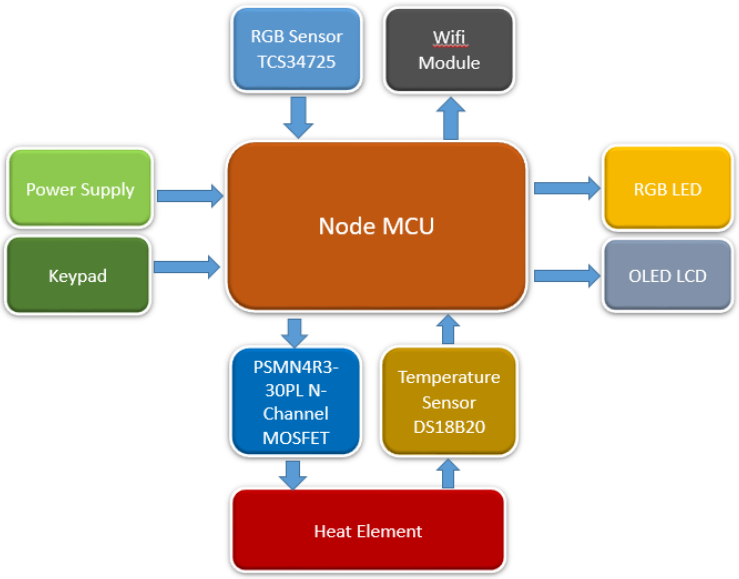


Gambar 1: Diagram deteksi Covid-19 realtime dan monitoring system

Aplikasi tombol darurat Covid-19 IoT adalah aplikasi berbasis android sederhana yang dibuat menggunakan Android Studio dan dirancang untuk digunakan oleh orang yang terpapar Covid-19 hanya dengan menekan tombol darurat pada aplikasi. Sesaat setelah tombol ditekan, aplikasi akan mengirimkan data diri pengguna dan data lokasi yang berupa latitude dan longitude menggunakan internet ke cloud database. iLAMP adalah perangkat pendeteksi Covid-19 yang terhubung ke cloud database menggunakan IoT. iLAMP ditempatkan di rumah sakit dan Puskesmas. Ketika iLAMP memberikan hasil positif, negatif maupun eror setelah melalui perbandingan data sampel maka, perangkat IoT yang terintegrasi dengan LAMP mengirim data diri pengguna dan lokasi ke cloud database. Cloud database yang dikirimkan oleh kedua aplikasi tersebut dapat diakses melalui website pangkalan data utama yang menampilkan lokasi di peta.

2.2. Hardware design

iLAMP dibuat berdasarkan proyek P-LAMP (The Pocket LAMP, 2021) dengan beberapa tambahan modifikasi. Perangkat keras LAMP terdiri dari NodeMCU V3 Lolin yang merupakan platform IoT yang bersifat opensource dengan pusat proccessing berupa System on Chip ESP8266. NodeMCU V3 Lolin juga telah memiliki kapabilitas akses terhadap WiFi untuk mengirim dan menerima data melalui konektivitas WiFi. NodeMCU terhubung dengan beberapa sensor, yaitu Red Green Blue (RGB) TC34725 sensor yang berfungsi untuk mendeteksi warna RGB suatu objek dan sensor suhu DS18B20 untuk mendeteksi panas yang dihasilkan oleh elemen pemanas (resistor) pada rangkaian seperti yang ditunjukkan pada gambar 2.

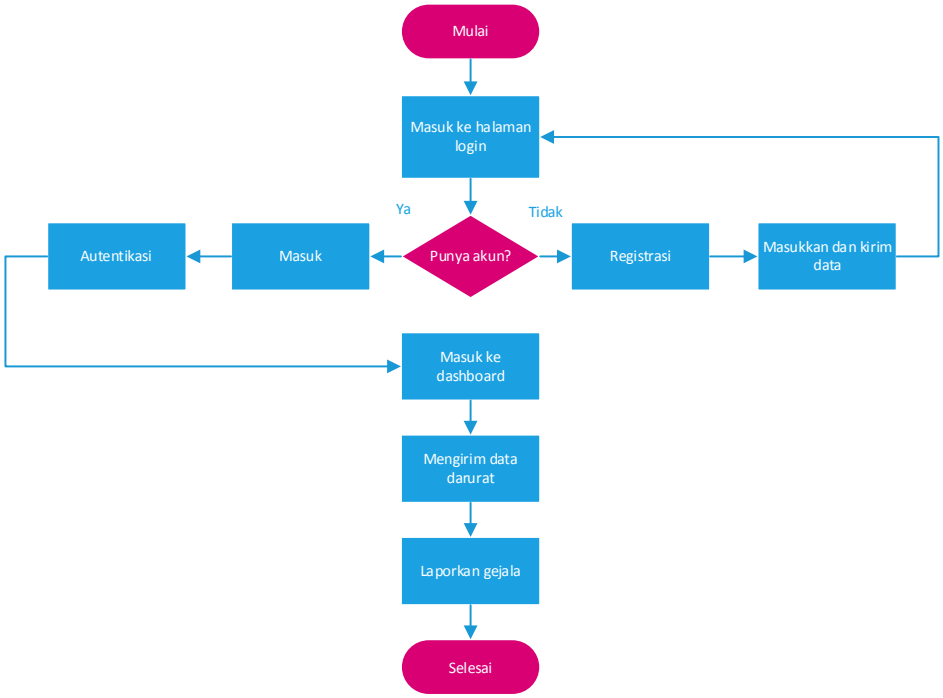


Gambar 2: Diagram iLAMP

Proses pemanasan ini dihasilkan dari penyaklaran tegangan 12V oleh MOSFET IRF3205 yang menyebabkan rangkaian arus pendek pada resistor. NodeMCU juga terhubung dengan LED RGB sebagai indikator untuk menandakan instruksi dan hasil pembacaan sensor. selain itu, NodeMCU juga dilengkapi OLED sebagai display Interface dalam bentuk teks dan gambar sehingga perangkat dapat dioperasikan dengan mudah oleh operator. Hasil pengukuran ini akan dikirim ke cloud database melalui internet menggunakan konektivitas WiFi yang terpasang pada Board NodeMCU.

2.3. Software design

Aplikasi tombol darurat Covid-19 IoT dibuat menggunakan Android Studio. Pembuatan aplikasi tersebut dimulai dengan membangun User Interface aplikasi yang menjembatani sistem perangkat lunak dengan pengguna. Selanjutnya aplikasi diprogram agar dapat mengirim dan menampilkan data dengan baik mengikuti flowchart berikut:

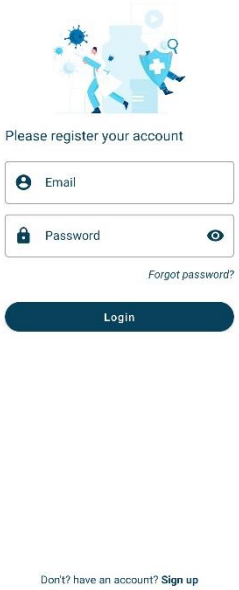


Gambar 3: Flowchart Aplikasi Tombol Darurat Covid-19

Pengguna mendaftar ke sistem dengan memasukkan data identitas pada formulir registrasi. Data tersebut akan di kirim ke database sebagai akses masuk ke dashboard aplikasi. Pengguna yang telah terdaftar dapat masuk ke aplikasi melalui proses autentikasi berdasarkan data yang telah di registrasikan sebelumnya. Jika proses autentikasi berhasil maka akan masuk ke halaman dashboard. Pengguna yang merasa dirinya terpapar covid-19 dengan gejala yang dialami maka pengguna dapat menekan tombol darurat. Tombol darurat akan mengirimkan informasi beserta lokasi realtime ke database. Selain tombol darurat dapat melaporkan gejala yang dialami. Data yang dikirim akan diproses oleh pihak kesehatan terkait.

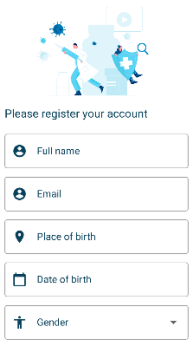
2. Result

Aplikasi tombol darurat Covid-19 IoT terdiri beberapa bagian. Bagian pertama merupakan halaman login untuk mengisi username dan password seperti yang ditunjukkan pada gambar 4.



Gambar 4: Halaman Login

Halaman ini bertujuan untuk memudahkan pengguna untuk masuk dan menggunakan aplikasi sesuai dengan data diri yang telah mereka daftarkan sewaktu-waktu ketika hendak menggunakan tombol darurat Covid-19. Bagian kedua merupakan formulir registrasi untuk mengisi data pribadi pengguna seperti yang ditunjukkan pada gambar 5.



Gambar 5: Halaman Registrasi

Halaman ini bertujuan untuk merekam identitas pengguna yang akan dikirim ke cloud database sewaktu-waktu pengguna menekan tombol darurat Covid-19. Bagian ketiga merupakan dashboard aplikasi yang memuat tombol darurat Covid-19 ditekan jika pengguna telah menginformasikan bahwa dirinya telah terpapar virus corona melalui pengujian yang dilakukan atau menyampaikan gejala yang diderita untuk dilaporkan sehingga dapat ditindaklanjuti dengan melakukan pengujian di lokasi alat yang telah terpasang alat iLAMP untuk mengetahui pengguna telah terpapar atau tidak,

selain itu halaman ini juga memuat petunjuk protokol kesehatan dan formulir laporan gejala seperti yang ditunjukkan pada gambar 6.

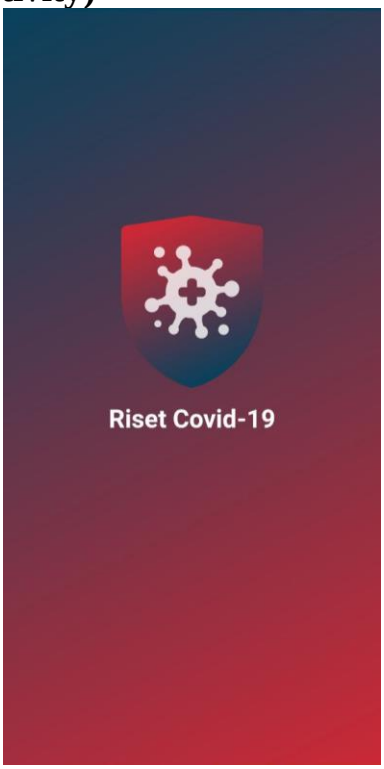
Gambar 6: Tampilan dashboard aplikasi

Bagian terakhir dari aplikasi merupakan halaman informasi yang memuat informasi terkait aplikasi ini seperti yang ditunjukkan pada gambar 7.

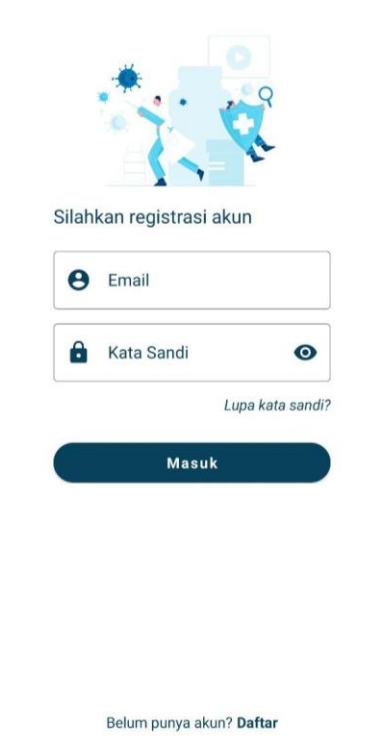
Gambar 7: halaman Informasi

TAMPILAN APLIKASI

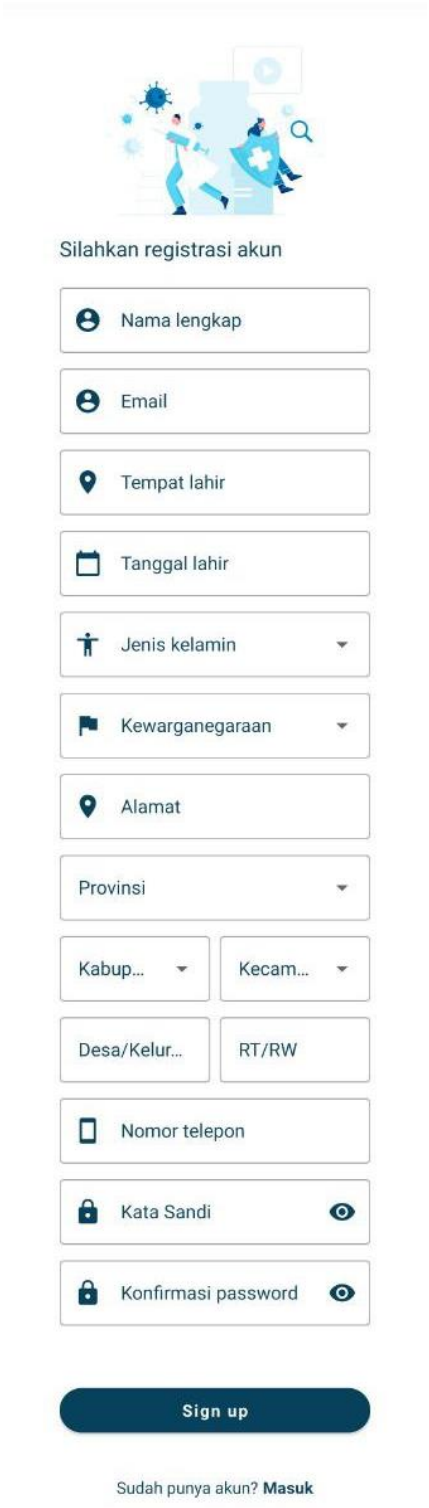
1. Splash Screen (Splash Activity)



2. Login Activity



3. Register Activity



4. Dashboard Activity

5. Information dan Logout Menu

Riset covid-19

Information

Logout

Geser kesamping >>

Apakah anda mengalami gejala dibawah?

- [ ] Suhu diatas 37ºC
- [ ] Batuk
- [ ] Sakit kepala
- [ ] Demam
- [ ] Sesak nafas
- [ ] Lemas
- [ ] Sembelit
- [ ] Sakit perut

Laporkan gejala

LISTING PROGRAM APLIKASI

1. Connection

a. Network

```
package com.hexatronik.risetcovid_19.network
import okhttp3.OkHttpClient
import okhttp3.logging.HttpLoggingInterceptor
import retrofit2.Retrofit
import retrofit2.adapter.rxjava3.RxJava3CallAdapterFactory
import retrofit2.converter.gson.GsonConverterFactory

object Network {
    fun getRetrofit(): Retrofit {
        val interceptor = HttpLoggingInterceptor()
        interceptor.level = HttpLoggingInterceptor.Level.BODY
        val client =
OkHttpClient.Builder().addInterceptor(interceptor).build()

        return Retrofit.Builder()
            .baseUrl("https://covidriset.000webhostapp.com/")
            .addConverterFactory(GsonConverterFactory.create())
            .addCallAdapterFactory(RxJava3CallAdapterFactory.create())
            .client(client)
            .build()
    }

    fun service(): Routes = getRetrofit().create(Routes::class.java)
}
```

b. Routes

```
package com.hexatronik.risetcovid_19.network

import com.hexatronik.risetcovid_19.model.emergency.EmergencyResponse
import com.hexatronik.risetcovid_19.model.report.ReportResponse
import com.hexatronik.risetcovid_19.model.user.UserResponse
import io.reactivex.rxjava3.core.Flowable
import retrofit2.http.Field
import retrofit2.http.FormUrlEncoded
import retrofit2.http.POST

interface Routes {
    // Login
    @FormUrlEncoded
    @POST("app_login.php")
    fun login(
        @Field("email") email: String,
        @Field("kata_sandi") password: String
    ): Flowable<UserResponse>

    // Register
    @FormUrlEncoded
    @POST("app_register.php")
    fun register(
        @Field("email") email: String,
```

```
        @Field("nama") name: String,
        @Field("tempat_lahir") pob: String,
        @Field("tanggal_lahir") dob: String,
        @Field("jenis_kelamin") gender: String,
        @Field("kewarganegaraan") citizenship: String,
        @Field("alamat") address: String,
        @Field("provinsi") province: String,
        @Field("kabupaten_kota") city: String,
        @Field("kecamatan") district: String,
        @Field("desa_kelurahan") village: String,
        @Field("rt_rw") rtRw: String,
        @Field("no_hp") phoneNumber: String,
        @Field("kata_sandi") password: String
    ): Flowable<UserResponse>

    // Laporan Gejala
    @FormUrlEncoded
    @POST("app_laporan.php")
    fun report(
        @Field("email") email: String,
        @Field("level_gejala") symptoms2: String,
        @Field("g_suhu") temp: String,
        @Field("g_batuk") cough: String,
        @Field("g_sakitkepala") headache: String,
        @Field("g_demam") fever: String,
        @Field("g_sesaknafas") oob: String,
        @Field("g_lemas") limp: String,
        @Field("g_sembelit") constipation: String,
        @Field("g_sakitperut") stomachache: String
    ): Flowable<ReportResponse>

    // Laporan Gejala
    @FormUrlEncoded
    @POST("app_darurat.php")
    fun emergency(
        @Field("email") email: String,
        @Field("level_gejala") symptoms: String
    ): Flowable<EmergencyResponse>

}
```

2. Model

a. Emergency

- **Emergency Item**

```
package com.hexatronik.risetcovid_19.model.emergency

import com.google.gson.annotations.SerializedName
data class EmergencyItem(
    @field:SerializedName("email")
    val email: String? = null,

    @field:SerializedName("level_gejala")
    val symptoms: String? = null
)
```

- **Emergency Response**

```
package com.hexatronik.risetcovid_19.model.emergency

import com.google.gson.annotations.SerializedName
data class EmergencyResponse(
    @field:SerializedName("data")
    val data: List<EmergencyItem>? = null,

    @field:SerializedName("message")
    val message: String? = null,

    @field:SerializedName("isSuccess")
    val isSuccess: Boolean? = null
)
```

b. Report

- **Report Item**

```
package com.hexatronik.risetcovid_19.model.report

import com.google.gson.annotations.SerializedName
data class ReportItem(
    @field:SerializedName("email")
    val email: String? = null,

    @field:SerializedName("level_gejala")
    val symptoms2: String? = null,

    @field:SerializedName("g_suhu")
    val temp: String? = null,

    @field:SerializedName("g_batuk")
    val cough: String? = null,

    @field:SerializedName("g_sakitkepala")
    val headache: String? = null,

    @field:SerializedName("g_demam")
    val fever: String? = null,

    @field:SerializedName("g_sesaknafas")
    val oob: String? = null,

    @field:SerializedName("g_lemas")
    val limp: String? = null,

    @field:SerializedName("g_sembelit")
    val constipation: String? = null,

    @field:SerializedName("g_sakitperut")
    val stomachache: String? = null,
)
```

- **Report Response**

```
package com.hexatronik.risetcovid_19.model.report

import com.google.gson.annotations.SerializedName
data class ReportResponse (
    @field:SerializedName("data")
    val data: List<ReportItem>? = null,

    @field:SerializedName("message")
    val message: String? = null,

    @field:SerializedName("isSuccess")
    val isSuccess: Boolean? = null
)
```

c. User

- **User Item**

```
package com.hexatronik.risetcovid_19.model.user

import com.google.gson.annotations.SerializedName
data class UserItem(

        @field:SerializedName("id")
        val id: String? = null,

        @field:SerializedName("email")
        val email: String? = null,

        @field:SerializedName("nama")
        val name: String? = null,

        @field:SerializedName("tempat_lahir")
        val pob: String? = null,

        @field:SerializedName("tanggal_lahir")
        val dob: String? = null,

        @field:SerializedName("jenis_kelamin")
        val gender: String? = null,

        @field:SerializedName("kewarganegaraan")
        val citizenship: String? = null,

        @field:SerializedName("alamat")
        val address: String? = null,

        @field:SerializedName("provinsi")
        val province: String? = null,

        @field:SerializedName("kecamatan")
        val district: String? = null,

        @field:SerializedName("kabupaten_kota")
        val city: String? = null,
```

```
        @field:SerializedName("desa_kelurahan")
        val village: String? = null,

        @field:SerializedName("rt_rw")
        val rtRw: String? = null,

        @field:SerializedName("no_hp")
        val phoneNumber: String? = null,

        @field:SerializedName("kata_sandi")
        val password: String? = null
)
```

- **User Response**

```
package com.hexatronik.risetcovid_19.model.user

import com.google.gson.annotations.SerializedName
data class UserResponse (
    @field:SerializedName("data")
    val data: List<UserItem>? = null,

    @field:SerializedName("message")
    val message: String? = null,

    @field:SerializedName("isSuccess")
    val isSuccess: Boolean? = null
)
```

3. UI (User Interface)

a. Splash

- **Splash Activity**

```
package com.hexatronik.risetcovid_19.ui.splash

import android.content.Intent
import android.os.Bundle
import android.os.Handler
import androidx.appcompat.app.AppCompatActivity
import
com.hexatronik.risetcovid_19.ui.main.dashboard.DashboardActivity
import com.hexatronik.risetcovid_19.R
import com.hexatronik.risetcovid_19.ui.login.LoginActivity
import com.hexatronik.risetcovid_19.utils.SessionManager

class SplashScreenActivity : AppCompatActivity() {
    val splashTime: Long = 500
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splash_screen)
        supportActionBar?.hide()

        val session = SessionManager(this)
        Handler().postDelayed(Runnable {
            if (session.login != false) {
                overridePendingTransition(R.anim.slide_up_in,
```

```
R.anim.slide_up_out)
                startActivity(Intent(this,
DashboardActivity::class.java))
            } else {

overridePendingTransition(R.anim.slide_up_in,R.anim.slide_up_out)
                startActivity(Intent(this,
LoginActivity::class.java))
            }
            finish()
        }, splashTime)
    }
}
```

b. Login

- Login Activity

```
package com.hexatronik.risetcovid_19.ui.login

import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.*
import androidx.appcompat.app.AppCompatActivity
import
com.hexatronik.risetcovid_19.ui.main.dashboard.DashboardActivity
import com.hexatronik.risetcovid_19.R
import com.hexatronik.risetcovid_19.model.user.UserItem
import com.hexatronik.risetcovid_19.ui.register.RegisterActivity
import com.hexatronik.risetcovid_19.utils.SessionManager

class LoginActivity : AppCompatActivity(), LoginView {
    private var presenter: LoginPresenter? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)
        presenter = LoginPresenter(this)
        supportActionBar?.hide()

        val btnLogin = findViewById<Button>(R.id.btn_login)
        val btnGotoRegister =
findViewById<TextView>(R.id.btn_goto_register)
        val etEmail = findViewById<EditText>(R.id.et_email_login)
        val etPassword =
findViewById<EditText>(R.id.et_password_login)


        btnGotoRegister.setOnClickListener {
            val i = Intent(applicationContext,
RegisterActivity::class.java)
            startActivity(i)
            finishAffinity()
        }

        btnLogin.setOnClickListener {
```

```
            val email = etEmail.text.toString()
            val password =  etPassword.text.toString()

            presenter?.loginUser(email, password)
        }
    }

    override fun showProgress() {
        val progress =
findViewById<ProgressBar>(R.id.progress_bar_login)
        val btnLogin = findViewById<Button>(R.id.btn_login)
        progress.visibility = View.VISIBLE
        btnLogin.visibility = View.GONE
    }

    override fun hideProgress() {
        val progress =
findViewById<ProgressBar>(R.id.progress_bar_login)
        val btnLogin = findViewById<Button>(R.id.btn_login)

        progress.visibility = View.GONE
        btnLogin.visibility = View.VISIBLE
    }

    override fun successLogin(msg: String, response:
List<UserItem>?) {
        val session = SessionManager(this)
        session.email = response?.get(0)?.email
        session.login = true

        Toast.makeText(this, "Login Berhasil",
Toast.LENGTH_SHORT).show()
        startActivity(Intent(this, DashboardActivity::class.java))
        finish()
    }

    override fun errorLogin(msg: String) {
        Toast.makeText(this, "Email atau Password Salah",
Toast.LENGTH_SHORT).show()
    }

    override fun onBackPressed() {
        super.onBackPressed()
        super.finishAffinity()
    }
}
```

- **Login Presenter**

```
package com.hexatronik.risetcovid_19.ui.login

import com.hexatronik.risetcovid_19.network.Network
import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers
import io.reactivex.rxjava3.schedulers.Schedulers
class LoginPresenter(val loginView: LoginView) {

    fun loginUser(
```

```
            email: String,
            password: String
    ) {
        loginView.showProgress()
        if (email.isNotEmpty() && password.isNotEmpty()) {
            val login = Network.service().login(email, password)
            login.subscribeOn(Schedulers.io())
                    .observeOn(AndroidSchedulers.mainThread())
                    .subscribe({ response ->
                        if (response.isSuccess != false) {
                            response.message?.let {
loginView.successLogin(it, response.data) }
                            loginView.hideProgress()
                        } else {
                            loginView.errorLogin(response.message ?:
"")
                            loginView.hideProgress()
                        }
                    }, { err ->
                        loginView.errorLogin(err.localizedMessage ?:
"")
                        loginView.hideProgress()
                    })
        } else {
            loginView.errorLogin("Tidak ada yang boleh kosong")
            loginView.hideProgress()
        }
    }
}
```

- **Login View**

```
package com.hexatronik.risetcovid_19.ui.login

import com.hexatronik.risetcovid_19.model.user.UserItem
interface LoginView {
    fun showProgress()
    fun hideProgress()
    fun successLogin(msg: String, response: List<UserItem>?)
    fun errorLogin(msg: String)
}
```

c. Register

- **Register Activity**

```
package com.hexatronik.risetcovid_19.ui.register

import android.app.DatePickerDialog
import android.app.DatePickerDialog.OnDateSetListener
import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.*
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.textfield.TextInputEditText
import com.hexatronik.risetcovid_19.R
import com.hexatronik.risetcovid_19.ui.login.LoginActivity
```

```
import kotlinx.android.synthetic.main.activity_register.*
import java.text.SimpleDateFormat
import java.util.*

class RegisterActivity : AppCompatActivity(), RegisterView {
    private var presenter: RegisterPresenter? = null
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_register)
        supportActionBar?.hide()

        presenter = RegisterPresenter(this)

        val myCalendar = Calendar.getInstance()
        val btnSignUp = findViewById<Button>(R.id.btn_sign_up)
        val btnBackLogin =
findViewById<TextView>(R.id.btn_back_login)
        val etName = findViewById<EditText>(R.id.et_name_register)
        val etEmail = findViewById<EditText>(R.id.et_email_register)
        val etPob = findViewById<EditText>(R.id.et_pob_register)
        val acDob =
findViewById<TextInputEditText>(R.id.ac_dob_register) as EditText
        val acGender =
findViewById<TextInputEditText>(R.id.ac_gender_register) as EditText
        val acCitizenship =
findViewById<EditText>(R.id.ac_citizenship_register)
        val etAddress =
findViewById<EditText>(R.id.et_address_register)
        val acProvince =
findViewById<EditText>(R.id.ac_province_register)
        val acCity = findViewById<EditText>(R.id.ac_city_register)
        val acDistrict =
findViewById<EditText>(R.id.ac_district_register)
        val etVillage =
findViewById<EditText>(R.id.ac_village_register)
        val etRtRw = findViewById<EditText>(R.id.et_rtrw_register)
        val etPhoneNumber =
findViewById<EditText>(R.id.et_phone_number_register)
        val etPassword =
findViewById<EditText>(R.id.et_password_register)
        val etConfirmPassword =
findViewById<EditText>(R.id.et_confirm_password_register)

        val date =
            OnDateSetListener { view, year, monthOfYear, dayOfMonth
-> /* TODO Auto-generated method stub */
                myCalendar[Calendar.YEAR] = year
                myCalendar[Calendar.MONTH] = monthOfYear
                myCalendar[Calendar.DAY_OF_MONTH] = dayOfMonth
                val myFormat = "dd/MM/yy" //In which you need put
here
                val sdf = SimpleDateFormat(myFormat, Locale.US)
                acDob.setText(sdf.format(myCalendar.time))
            }

        acDob.setOnClickListener { /* TODO Auto-generated method
stub */
```

```
            DatePickerDialog(
                this@RegisterActivity, date,
                myCalendar[Calendar.YEAR],
                myCalendar[Calendar.MONTH],
                myCalendar[Calendar.DAY_OF_MONTH]
            ).show()
        }

        btnBackLogin.setOnClickListener {
            val i = Intent(applicationContext,
LoginActivity::class.java)
            startActivity(i)
            finishAffinity()
        }

        btnSignUp.setOnClickListener {
            val name = etName.text.toString()
            val email = etEmail.text.toString()
            val pob = etPob.text.toString()
            val dob = acDob.text.toString()
            val gender = acGender.text.toString()
            val citizenship = acCitizenship.text.toString()
            val address = etAddress.text.toString()
            val province = acProvince.text.toString()
            val city = acCity.text.toString()
            val district = acDistrict.text.toString()
            val village = etVillage.text.toString()
            val rtRw = etRtRw.text.toString()
            val phoneNumber = etPhoneNumber.text.toString()
            val password = etPassword.text.toString()
            val confirmPassword = etConfirmPassword.text.toString()

            //Toast.makeText(this, email, Toast.LENGTH_SHORT).show()

            presenter?.registerUser(
                name, email, pob, dob, gender, citizenship, address,
province, city, district, village, rtRw, phoneNumber, password,
confirmPassword
            )
        }

        /* Dropdown Jenis kelamin */
        val itemJenisKelamin =
this.findViewById<AutoCompleteTextView>(R.id.ac_gender_register)
        val jenisKelamin =
resources.getStringArray(R.array.jenis_kelamin)
        val arrayAdapterJenisKelamin = ArrayAdapter(this,
R.layout.dropdown_jeniskelamin, jenisKelamin)
        itemJenisKelamin.setAdapter(arrayAdapterJenisKelamin)

        /* Dropdown Kewarganegaraan */
        val itemKewarganegaraan =
this.findViewById<AutoCompleteTextView>(R.id.ac_citizenship_register
)
        val kewarganegaraan =
resources.getStringArray(R.array.kewarganegaraan)
        val arrayAdapterKewargangeraan = ArrayAdapter(this,
```

```
R.layout.dropdown_jeniskelamin, kewarganegaraan)
        itemKewarganegaraan.setAdapter(arrayAdapterKewargangeraan)

        /* Dropdown Provinsi */
        val itemProvinsi =
this.findViewById<AutoCompleteTextView>(R.id.ac_province_register)
        val provinsi = resources.getStringArray(R.array.provinsi)
        val arrayAdapterProvinsi = ArrayAdapter(this,
R.layout.dropdown_jeniskelamin, provinsi)
        itemProvinsi.setAdapter(arrayAdapterProvinsi)

        /* Dropdown Kabupaten */
        val itemKabupaten =
this.findViewById<AutoCompleteTextView>(R.id.ac_city_register)
        val kabupaten = resources.getStringArray(R.array.kabupaten)
        val arrayAdapterKabupaten = ArrayAdapter(this,
R.layout.dropdown_jeniskelamin, kabupaten)
        itemKabupaten.setAdapter(arrayAdapterKabupaten)

        /* Dropdown Kecamatan */
        val itemKecamatan =
this.findViewById<AutoCompleteTextView>(R.id.ac_district_register)
        val kecamatan = resources.getStringArray(R.array.kecamatan)
        val arrayAdapterKecamatan = ArrayAdapter(this,
R.layout.dropdown_jeniskelamin, kecamatan)
        itemKecamatan.setAdapter(arrayAdapterKecamatan)

    }

    override fun showProgress() {
        val btnSignUp = findViewById<Button>(R.id.btn_sign_up)
        val progress =
findViewById<ProgressBar>(R.id.progress_bar_register)
        progress.visibility = View.VISIBLE
        btnSignUp.visibility = View.GONE
    }

    override fun hideProgress() {
        val btnSignUp = findViewById<Button>(R.id.btn_sign_up)
        val progress =
findViewById<ProgressBar>(R.id.progress_bar_register)
        progress.visibility = View.GONE
        btnSignUp.visibility = View.VISIBLE
    }

    override fun successRegister(msg: String) {
        startActivity(Intent(this, LoginActivity::class.java))
        finish()
    }

    override fun errorRegister(msg: String) {
        showToast(msg)
    }

    private fun showToast(msg: String) {
        Toast.makeText(this, msg, Toast.LENGTH_SHORT).show()
    }
```

```
    override fun onBackPressed() {
        super.onBackPressed()
        super.finishAffinity()
    }

    override fun onResume() {
        super.onResume()
        /* Dropdown Jenis kelamin */
    }
}
```

- **Register Presenter**

```
package com.hexatronik.risetcovid_19.ui.register

import com.hexatronik.risetcovid_19.network.Network
import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers
import io.reactivex.rxjava3.schedulers.Schedulers

class RegisterPresenter(private val registerView: RegisterView) {
    fun registerUser(
        name: String,
        email: String,
        pob: String,
        dob: String,
        gender: String,
        citizenship: String,
        address: String,
        province: String,
        city: String,
        district: String,
        village: String,
        rtRw: String,
        phoneNumber: String,
        password: String,
        confirmPassword: String
    ) {
        registerView.showProgress()
        if (name.isNotEmpty() &&
            email.isNotEmpty() &&
            pob.isNotEmpty() &&
            dob.isNotEmpty() &&
            gender.isNotEmpty() &&
            citizenship.isNotEmpty() &&
            address.isNotEmpty() &&
            province.isNotEmpty() &&
            city.isNotEmpty() &&
            district.isNotEmpty() &&
            village.isNotEmpty() &&
            phoneNumber.isNotEmpty() &&
            password.isNotEmpty()
        ){
            if (password != confirmPassword) {
                registerView.errorRegister("Password tidak cocok")
                registerView.hideProgress()
            } else if (password.length < 6) {
```

```
                registerView.errorRegister("Password tidak boleh
kurang dari 5")
                registerView.hideProgress()
            } else {
                val register = Network.service().register(name,
email, pob, dob, gender, citizenship, address, province, city,
district, village, rtRw, phoneNumber, password)
                register.subscribeOn(Schedulers.io())
                    .observeOn(AndroidSchedulers.mainThread())
                    .subscribe({ response ->
                        if (response.isSuccess != false) {
                            if (response.message == "Registrasi
Berhasil") {

registerView.successRegister(response.message)
                                registerView.hideProgress()
                            } else {

registerView.errorRegister(response.message ?: "")
                                registerView.hideProgress()
                            }
                        }
                    }, { err ->

registerView.errorRegister(err.localizedMessage ?: "")
                        registerView.hideProgress()
                    })
            }
        } else {
            registerView.errorRegister("Mohon lengkapi data")
            registerView.hideProgress()
        }
    }
}
```

- **Register View**

```
package com.hexatronik.risetcovid_19.ui.register

interface RegisterView {
    fun showProgress()
    fun hideProgress()
    fun successRegister(msg: String)
    fun errorRegister(msg: String)
}
```

d. Main

- **Dashboard**
  - **Dashboard Activity**

```
package com.hexatronik.risetcovid_19.ui.main.dashboard

import android.content.Intent
import android.os.Bundle
import android.view.Menu
import android.view.MenuItem
```

```
import android.view.View
import android.widget.*
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import
com.google.android.material.dialog.MaterialAlertDialogBuilder
import
com.hexatronik.risetcovid_19.ui.main.information.InformationActivi
ty
import com.hexatronik.risetcovid_19.R
import com.hexatronik.risetcovid_19.ui.login.LoginActivity
import com.hexatronik.risetcovid_19.utils.SessionManager

class DashboardActivity : AppCompatActivity(), DashboardView {
    private var presenter: DashboardPresenter? = null

    var mTemp: String? = null
    var mCough: String? = null
    var mHeadache: String? = null
    var mFever: String? = null
    var mOob: String? = null
    var mLimp: String? = null
    var mConstipation: String? = null
    var mStomachache: String? = null
    var userEmail: String? = null
    var symptoms = "darurat"
    var symptoms2 = "sedang"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_dashboard)
        supportActionBar?.title = "Riset covid-19"
        presenter = DashboardPresenter(this)
        val session = SessionManager(this)

        userEmail = session.email.toString()

        val user = findViewById<TextView>(R.id.tv_user)
        val temp =
findViewById<CheckedTextView>(R.id.item_gejala1)
        val cough =
findViewById<CheckedTextView>(R.id.item_gejala2)
        val headache =
findViewById<CheckedTextView>(R.id.item_gejala3)
        val fever =
findViewById<CheckedTextView>(R.id.item_gejala4)
        val oob = findViewById<CheckedTextView>(R.id.item_gejala5)
        val limp =
findViewById<CheckedTextView>(R.id.item_gejala6)
        val constipation =
findViewById<CheckedTextView>(R.id.item_gejala7)
        val stomachache =
findViewById<CheckedTextView>(R.id.item_gejala8)

        val btnEmergency =
findViewById<ImageView>(R.id.btn_emergency)
        val btnReport = findViewById<Button>(R.id.btn_report)
```

```
        user.text = userEmail.toString()

        btnEmergency.setOnClickListener {
            MaterialAlertDialogBuilder(this)

.setTitle(resources.getString(R.string.dialog_emergency_text))

.setMessage(resources.getString(R.string.dialog_report_supporting_
text))
                .setNegativeButton("Batalkan") { dialog, which ->
                    // Respond to negative button press
                }
                .setPositiveButton("Yakin") { dialog, which ->
                    // Call we care dialog
                    val view =
                        View.inflate(this@DashboardActivity,
R.layout.dialog_we_care_view, null)
                    val builder =
AlertDialog.Builder(this@DashboardActivity)
                    builder.setView(view)
                    val dialog = builder.create()
                    dialog.show()

dialog.window?.setBackgroundDrawableResource(android.R.color.trans
parent)

                    presenter?.emergencyBtn(
                        userEmail.toString(),
                        symptoms
                    )

                    // Make Toast
                    Toast.makeText(this, "Laporan anda telah
dikirim", Toast.LENGTH_SHORT).show()


                }
                .show()
        }


        temp.setOnClickListener {
            temp.isChecked = !temp.isChecked
            mTemp = if (temp.isChecked){
                "Ya"
            } else {
                "Tidak"
            }
        }

        cough.setOnClickListener {
            cough.isChecked = !cough.isChecked
            mCough = if (cough.isChecked){
                "Ya"
            } else {
                "Tidak"
```

```
        }
    }

    headache.setOnClickListener {
        headache.isChecked = !headache.isChecked
        mHeadache = if (headache.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }

    fever.setOnClickListener {
        fever.isChecked = !fever.isChecked
        mFever = if (fever.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }

    oob.setOnClickListener {
        oob.isChecked = !oob.isChecked
        mOob = if (oob.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }

    limp.setOnClickListener {
        limp.isChecked = !limp.isChecked
        mLimp = if (limp.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }

    constipation.setOnClickListener {
        constipation.isChecked = !constipation.isChecked
        mConstipation = if (constipation.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }

    stomachache.setOnClickListener {
        stomachache.isChecked = !stomachache.isChecked
        mStomachache = if (stomachache.isChecked){
            "Ya"
        } else {
            "Tidak"
        }
    }
```

```
        btnReport.setOnClickListener {
            MaterialAlertDialogBuilder(this)

.setTitle(resources.getString(R.string.dialog_report_tittle))

.setMessage(resources.getString(R.string.dialog_report_supporting_
text))
                .setNegativeButton("Batalkan") { dialog, which ->
                    // Respond to negative button press
                }
                .setPositiveButton("Yakin") { dialog, which ->
                    // Call we care dialog
                    val view =
                        View.inflate(this@DashboardActivity,
R.layout.dialog_we_care_view, null)
                    val builder =
AlertDialog.Builder(this@DashboardActivity)
                    builder.setView(view)
                    val dialog = builder.create()
                    dialog.show()

dialog.window?.setBackgroundDrawableResource(android.R.color.trans
parent)

                    presenter?.reportBtn(
                        userEmail.toString(),
                        symptoms2,
                        mTemp.toString(),
                        mCough.toString(),
                        mHeadache.toString(),
                        mFever.toString(),
                        mOob.toString(),
                        mLimp.toString(),
                        mConstipation.toString(),
                        mStomachache.toString()
                    )

                    // Make Toast
                    Toast.makeText(this, "Laporan anda telah
dikirim", Toast.LENGTH_SHORT).show()
                }
                .show()
        }

    }

    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.dashboard_menu, menu)
        return true
    }

    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        when (item.itemId) {
            R.id.information -> {
                val i = Intent(applicationContext,
InformationActivity::class.java)
                startActivity(i)
```

```
            }
            R.id.logout -> {
                val session = SessionManager(this)
                session.logOut()
                Toast.makeText(this, "Logout Berhasil",
Toast.LENGTH_SHORT).show()
                val intent = Intent(this,
LoginActivity::class.java)
                intent.flags = Intent.FLAG_ACTIVITY_NEW_TASK
                intent.flags = Intent.FLAG_ACTIVITY_CLEAR_TASK
                intent.flags = Intent.FLAG_ACTIVITY_CLEAR_TOP
                startActivity(intent)
                finish()
            }
        }
        return super.onOptionsItemSelected(item)
    }

    private fun showToast(msg: String) {
        //Toast.makeText(this, msg, Toast.LENGTH_LONG).show()
    }

    override fun showProgress() {
        //progress.visibility = View.VISIBLE
    }

    override fun hideProgress() {
        //progress.visibility = View.GONE
    }

    override fun onSuccess(msg: String) {
        //showToast(msg)
        //finish()
    }

    override fun onError(msg: String) {
        //showToast(msg)
    }
}
```

- **Dashboard Presenter**

```
package com.hexatronik.risetcovid_19.ui.main.dashboard

import com.hexatronik.risetcovid_19.network.Network
import io.reactivex.rxjava3.android.schedulers.AndroidSchedulers
import io.reactivex.rxjava3.schedulers.Schedulers

class DashboardPresenter(val dashboardView : DashboardView) {

    fun emergencyBtn(
        email: String,
        symptoms: String
    ) {
        dashboardView.showProgress()
        if (email.isNotEmpty() && symptoms.isNotEmpty()) {
            val inputEmergency = Network.service().emergency(
```

```
                email,
                symptoms
            )
            inputEmergency.subscribeOn(Schedulers.io())
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe({ response ->
                    if (response.isSuccess != false) {
                        if (response.message == "Laporan Telah
Dikirim") {

dashboardView.onSuccess(response.message)
                            dashboardView.hideProgress()
                        } else {
                            dashboardView.onError(response.message
?: "")
                            dashboardView.hideProgress()
                        }
                    }
                }, { err ->
                    dashboardView.onError(err.localizedMessage ?:
"")
                    dashboardView.hideProgress()
                })
        } else {
            dashboardView.onError("Kolom tidak boleh kosong")
            dashboardView.hideProgress()
        }
    }


    fun reportBtn(
        email: String,
        symptoms2: String,
        temp: String,
        cough: String,
        headache: String,
        fever: String,
        oob: String,
        limp: String,
        constipation: String,
        stomachache: String
    ) {
        dashboardView.showProgress()
        if (email.isNotEmpty() && symptoms2.isNotEmpty()) {
            val inputReport = Network.service().report(
                email,
                symptoms2,
                temp,
                cough,
                headache,
                fever,
                oob,
                limp,
                constipation,
                stomachache
            )
            inputReport.subscribeOn(Schedulers.io())
```

```
                .observeOn(AndroidSchedulers.mainThread())
                .subscribe({ response ->
                    if (response.isSuccess != false) {
                        if (response.message == "Laporan Telah
Dikirim") {

dashboardView.onSuccess(response.message)
                            dashboardView.hideProgress()
                        } else {
                            dashboardView.onError(response.message
?: "")
                            dashboardView.hideProgress()
                        }
                    }
                }, { err ->
                    dashboardView.onError(err.localizedMessage ?:
"")
                    dashboardView.hideProgress()
                })
        } else {
            dashboardView.onError("Kolom tidak boleh kosong")
            dashboardView.hideProgress()
        }
    }
}
```

- **Dashboard View**

```
package com.hexatronik.risetcovid_19.ui.main.dashboard

interface DashboardView {
    fun showProgress()
    fun hideProgress()
    fun onSuccess(msg: String)
    fun onError(msg: String)
}
```

- **Information**
  - **Information Activity**

```
package com.hexatronik.risetcovid_19.ui.main.information

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import com.hexatronik.risetcovid_19.R

class InformationActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_information)
        supportActionBar?.title="Informasi"


    }
}
```

4. Utils

a. Session Manager

```
package com.hexatronik.risetcovid_19.utils

import android.content.SharedPreferences
import android.content.Context

class SessionManager(context: Context) {
    private var pref: SharedPreferences? = null
    private var editor: SharedPreferences.Editor? = null
    var PREF_NAME = "RisetCovid19"

    var ISLOGIN = "isLogin"
    var NAME = "name"
    var EMAIL = "email"
    var PHONE = "phone"
    var TIME = "time"

    init {
        pref = context.getSharedPreferences(PREF_NAME, 0)
        editor = pref?.edit()
    }

    var login: Boolean?
        get() = pref?.getBoolean(ISLOGIN, false)
        set(login) {
            editor?.putBoolean(ISLOGIN, true)
            editor?.commit()
        }

    var email: String?
        get() = pref?.getString(EMAIL, "")
        set(email) {
            editor?.putString(EMAIL, email)
            editor?.commit()
        }

    fun logOut() {
        editor?.clear()
        editor?.commit()
    }
}
```