LAPORAN PRAKTIK UAS APLIKASI MANAJEMEN TOKO SEMBAKO

DISUSUN OLEH :

Aditya Pratama	2105102001
Efrhillia Windy	2105102011
Israminaza Ilma Redanda	2105102033

Kelas	: MI-5A
Mata Kuliah	: Praktik Pem. Mobile Lanjut

PROGRAM STUDI MANAJEMEN INFORMATIKA
JURUSAN TEKNIK KOMPUTER DAN INFORMATIKA
POLITEKNIK NEGERI MEDAN 2023

KATA PENGANTAR

Puji syukur bagi Tuhan Yang Maha Esa yang telah memberikan nikmat serta hidayah-Nya sehingga penyusun dapat menyelesaikan Laporan Praktik yang berjudul "APLIKASI MANAJEMEN TOKO SEMBAKO". Paper ini disusun bertujuan untuk memenuhi salah tugas mata kuliah Praktek Pemprograman Web.

Terima kasih saya ucapkan kepada Bapak sebagai Dosen Pengampu Praktik Pemrograman Mobile Lanjut yang telah membantu kami secara materi. Terima kasih juga kami ucapkan kepada teman—teman yang telah mendukung kami sehingga bisa menyelesaikan tugas ini tepat waktu. Kami menyadari, bahwa laporan yang kami buat ini masih jauh dari kata sempurna. Oleh karena itu, kami sangat menerima kritik dan saran yang membangun agar bisa menjadi lebih baik lagi dimasa mendatang.

Semoga laporan ini bisa menambah wawasan para pembaca dan bermanfaat untuk perkembangan dan peningkatan ilmu pengetahuan.

Medan, 23 Desember 2023

Penulis

DAFTAR ISI

BAB I PENDAHULUAN

1.1 Latar Belakang Masalah

Di era modern ini, penggunaan teknologi informasi telah menjadi elemen utama dalam perkembangan berbagai bisnis. Salah satu contoh bisnis yang telah mengadopsi teknologi untuk mengoptimalkan operasionalnya adalah bisnis toko sembako. Toko sembako memiliki peran yang sangat vital dalam memenuhi kebutuhan sehari-hari masyarakat, terutama dalam menyediakan produk-produk kebutuhan pokok seperti beras, minyak goreng, gula, dan produk sejenisnya.

Dalam menghadapi tantangan ini, pengembangan aplikasi manajemen toko sembako dapat menjadi solusi yang sangat berguna. Aplikasi semacam ini dapat membantu pemilik toko dalam manajemen persediaan, pencatatan penjualan dan keuangan, pelayanan pelanggan, analisis data, serta menjaga keamanan data. Dengan aplikasi ini, pemilik toko sembako dapat mengoptimalkan operasional, meningkatkan efisiensi, dan mengambil keputusan yang lebih baik berdasarkan informasi yang akurat.

Dalam makalah ini, kami akan mendiskusikan lebih lanjut tentang pentingnya aplikasi manajemen toko sembako, potensi manfaat yang dapat diperoleh, serta beberapa fitur kunci yang dapat diimplementasikan dalam aplikasi semacam ini. Tujuan dari makalah ini adalah memberikan pemahaman yang lebih mendalam tentang bagaimana teknologi informasi dapat memberikan dampak positif pada bisnis toko sembako dan memberikan panduan bagi pemilik toko yang ingin mempertimbangkan penggunaan aplikasi manajemen toko sembako dalam operasional mereka. Maka dari itu kami tertarik untuk membuat Tugas UAS Praktik Pemrograman Aplikasi Mobile

Lanjut yang berjudul **“Manajemen Toko Sembako”**

1.2 Rumusan Masalah

Berdasarkan latar belakang yang dikemukakan, rumusan masalah dari tugas akhir ini, yaitu:

1. Bagaimana penggunaan aplikasi manajemen dapat meningkatkan efisiensi operasional dalam toko sembako?
2. Bagaimana aplikasi manajemen dapat membantu dalam pengelolaan stok dan persediaan di toko sembako?
3. Apa saja fitur yang dibutuhkan dalam aplikasi manajemen toko sembako?

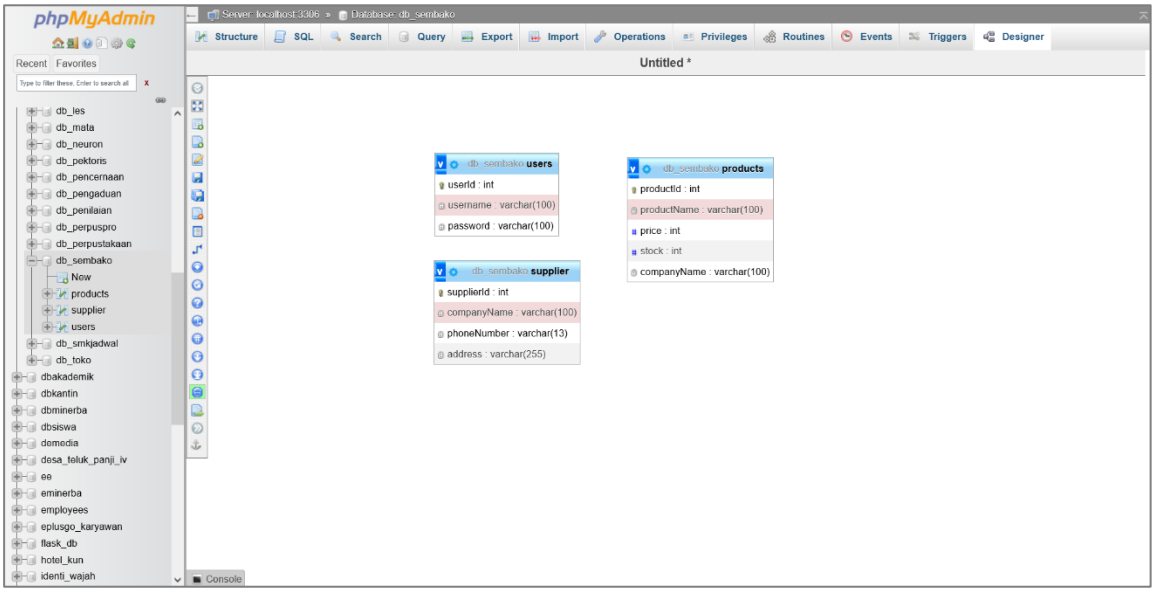
1.3 Tujuan

Berdasarkan Latar Belakang dan Rumusan Masalah dapat diketahui tujuannya. Tujuannya adalah sebagai berikut :

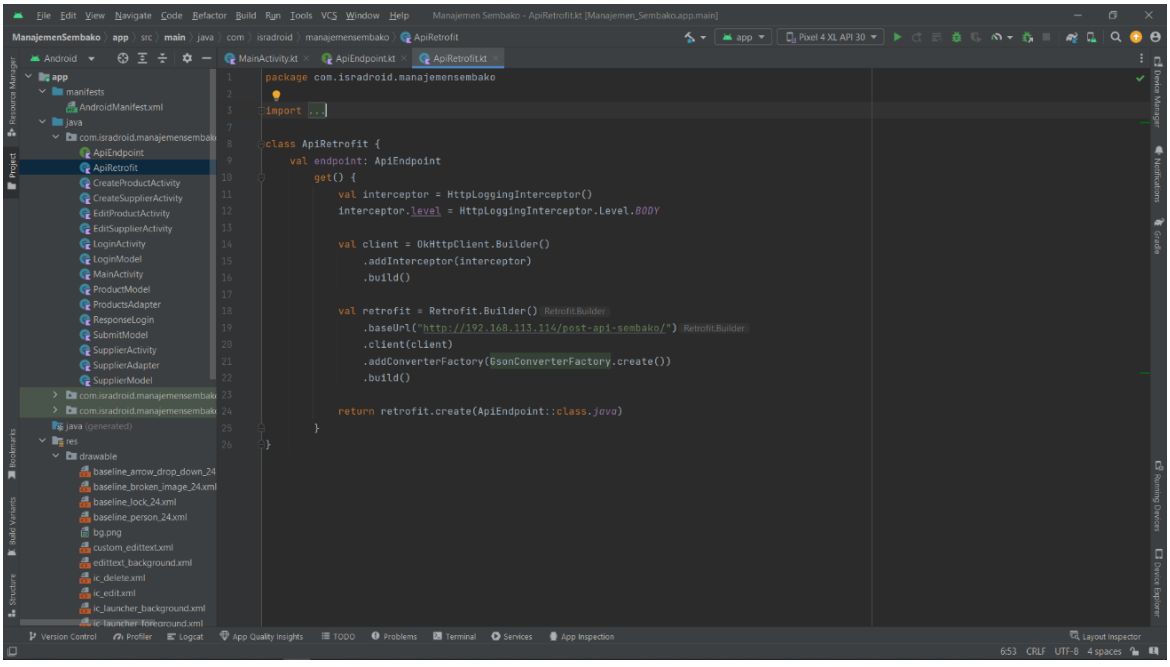
1. Memanfaatkan aplikasi manajemen toko untuk meningkatkan efisiensi dan efektivitas operasional toko sembako.
2. Mengelola stok barang dan persediaan secara akurat untuk mendukung kelancaran bisnis.
3. Mempermudah proses pencatatan penjualan, keuangan, dan pelayanan pelanggan dengan fitur-fitur yang relevan.

BAB II PEMBAHASAN

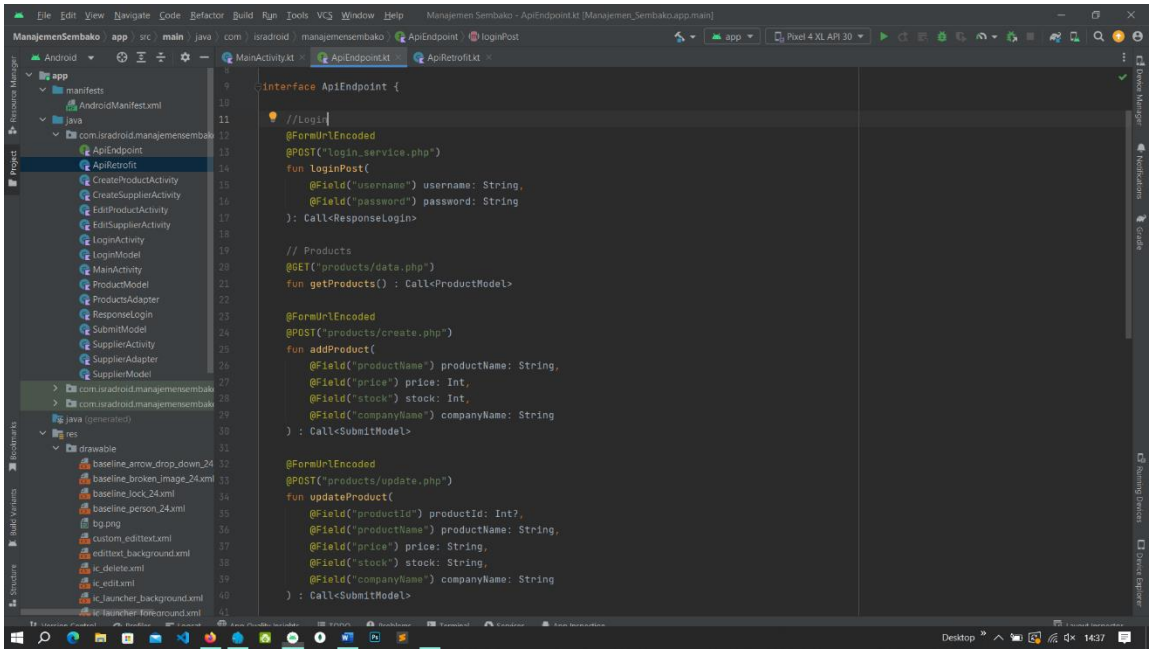
2.1 Proses Pembuatan Aplikasi dan CRUD



Gambar 2.1.1 Design Database db_sembako

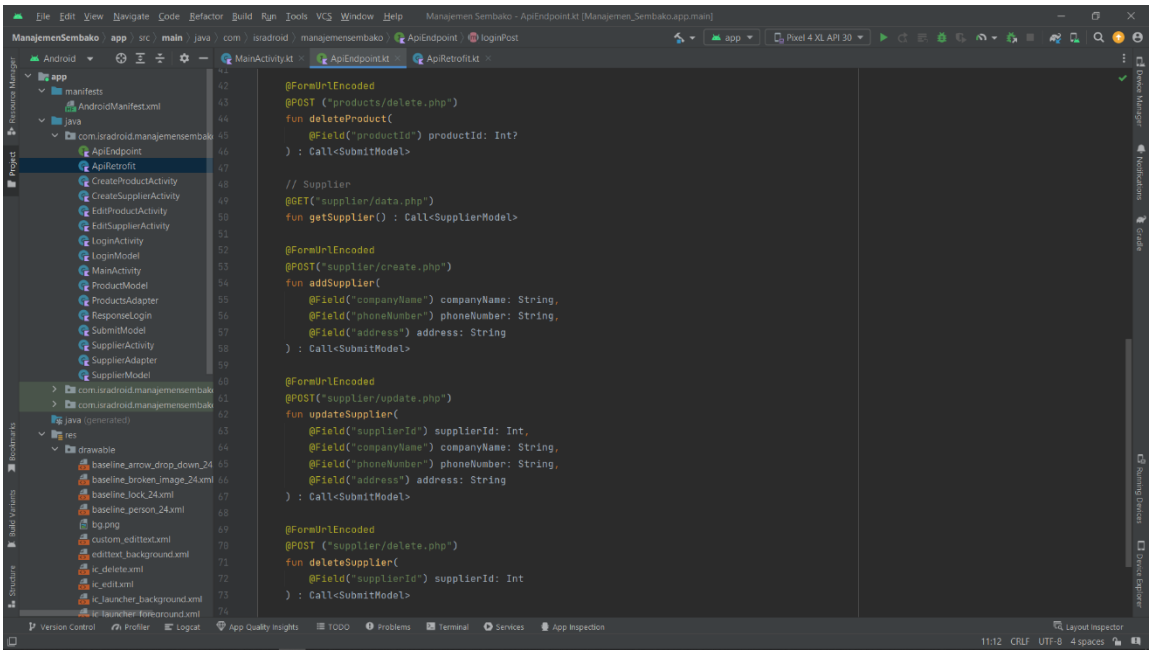


Gambar 2.1.2 Konfigurasi API PHP menggunakan Retrofit



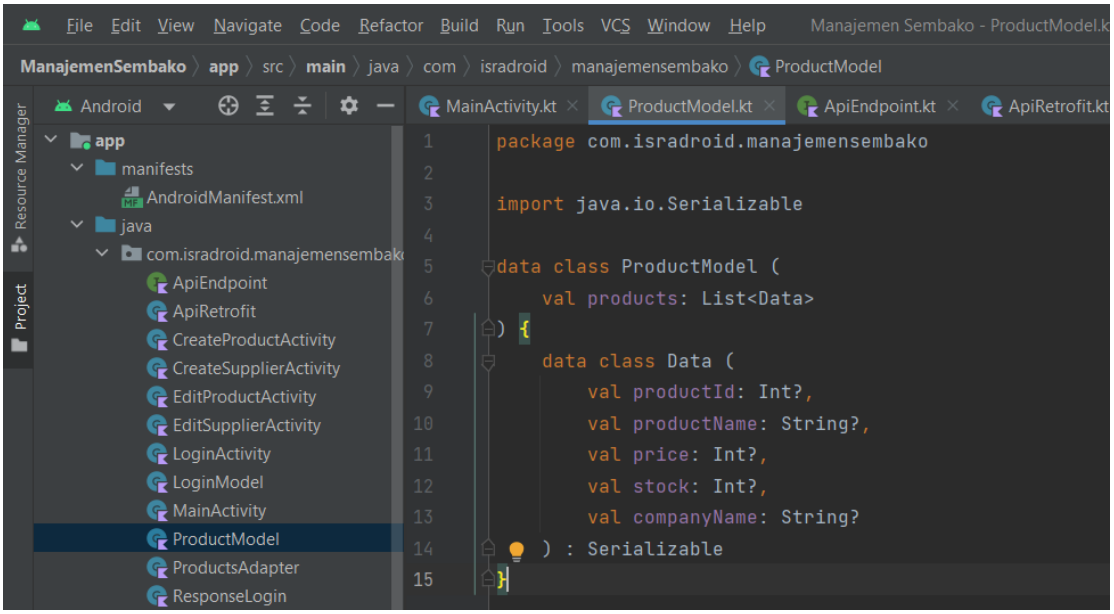
```
interface ApiEndpoint {

    //Login
    @FormUrlEncoded
    @POST("login_service.php")
    fun loginPost(
        @Field("username") username: String,
        @Field("password") password: String
    ): Call<ResponseLogin>

    // Products
    @GET("products/data.php")
    fun getProducts() : Call<ProductModel>

    @FormUrlEncoded
    @POST("products/create.php")
    fun addProduct(
        @Field("productName") productName: String,
        @Field("price") price: Int,
        @Field("stock") stock: Int,
        @Field("companyName") companyName: String
    ) : Call<SubmitModel>

    @FormUrlEncoded
    @POST("products/update.php")
    fun updateProduct(
        @Field("productId") productId: Int?,
        @Field("productName") productName: String,
        @Field("price") price: String,
        @Field("stock") stock: String,
        @Field("companyName") companyName: String
    ) : Call<SubmitModel>
```

Gambar 2.1.3 Membuat interface dari setiap endpoint

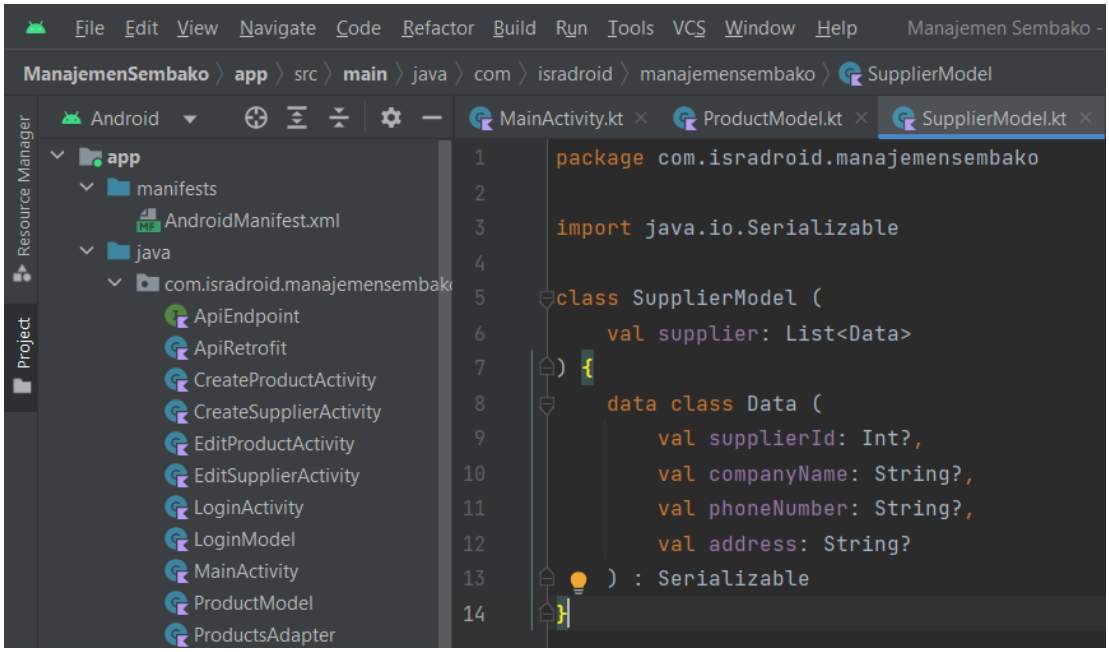


```
    @FormUrlEncoded
    @POST ("products/delete.php")
    fun deleteProduct(
        @Field("productId") productId: Int?
    ) : Call<SubmitModel>

    // Supplier
    @GET("supplier/data.php")
    fun getSupplier() : Call<SupplierModel>

    @FormUrlEncoded
    @POST("supplier/create.php")
    fun addSupplier(
        @Field("companyName") companyName: String,
        @Field("phoneNumber") phoneNumber: String,
        @Field("address") address: String
    ) : Call<SubmitModel>

    @FormUrlEncoded
    @POST("supplier/update.php")
    fun updateSupplier(
        @Field("supplierId") supplierId: Int,
        @Field("companyName") companyName: String,
        @Field("phoneNumber") phoneNumber: String,
        @Field("address") address: String
    ) : Call<SubmitModel>

    @FormUrlEncoded
    @POST ("supplier/delete.php")
    fun deleteSupplier(
        @Field("supplierId") supplierId: Int
    ) : Call<SubmitModel>
```

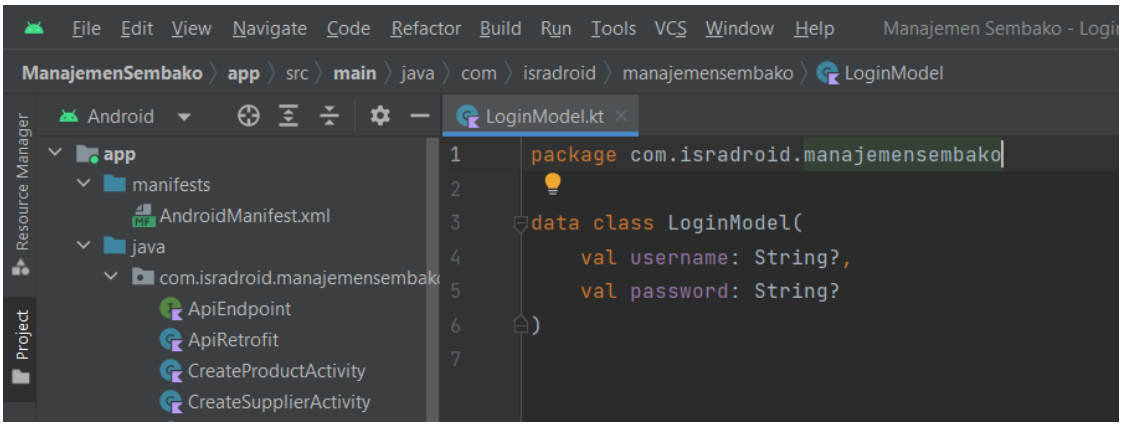
Gambar 2.1.4 Membuat interface dari setiap endpoint



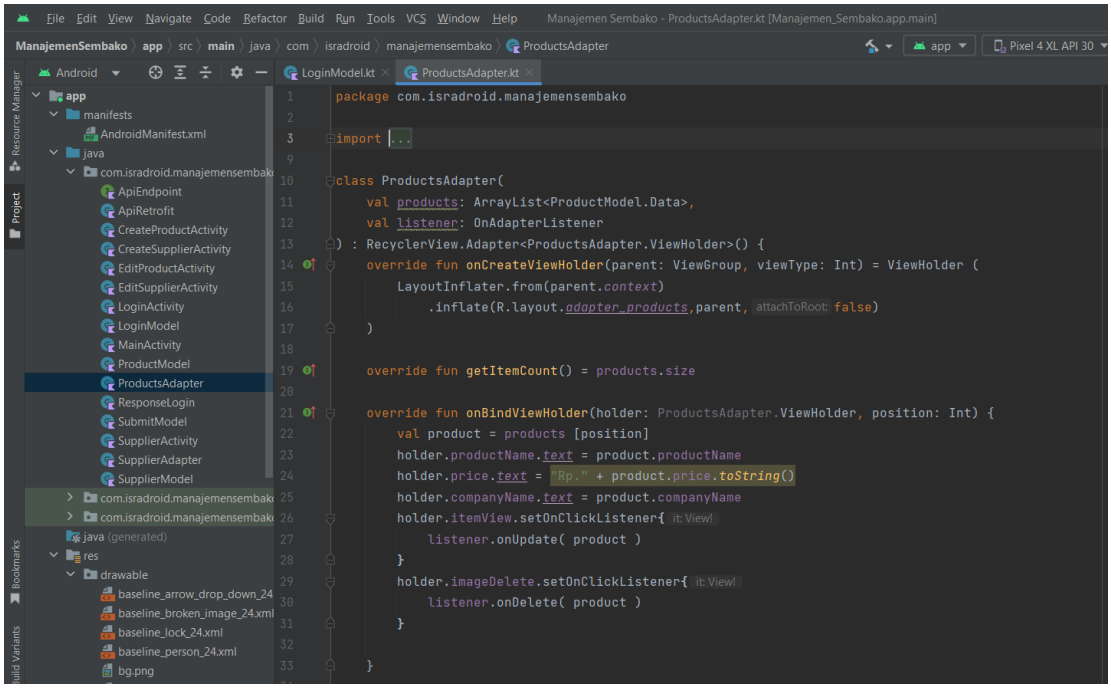
Gambar 2.1.4 Class ProductModel



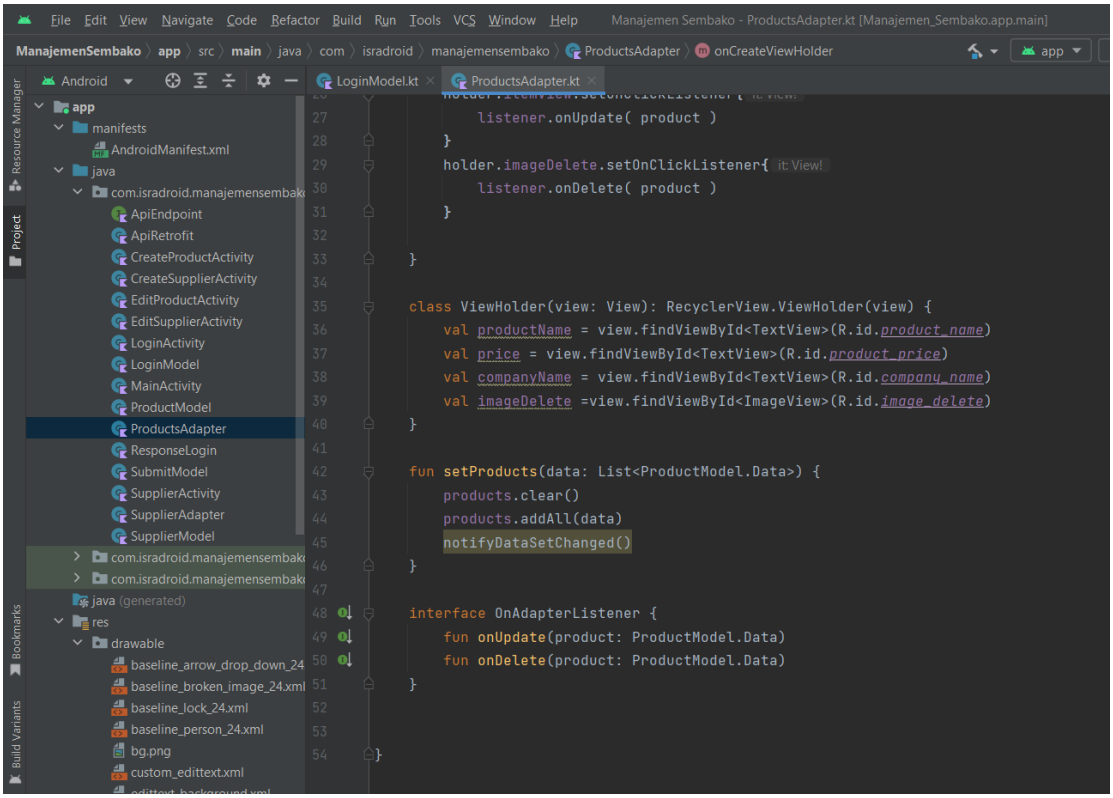
Gambar 2.1.5 Class SupplierModel



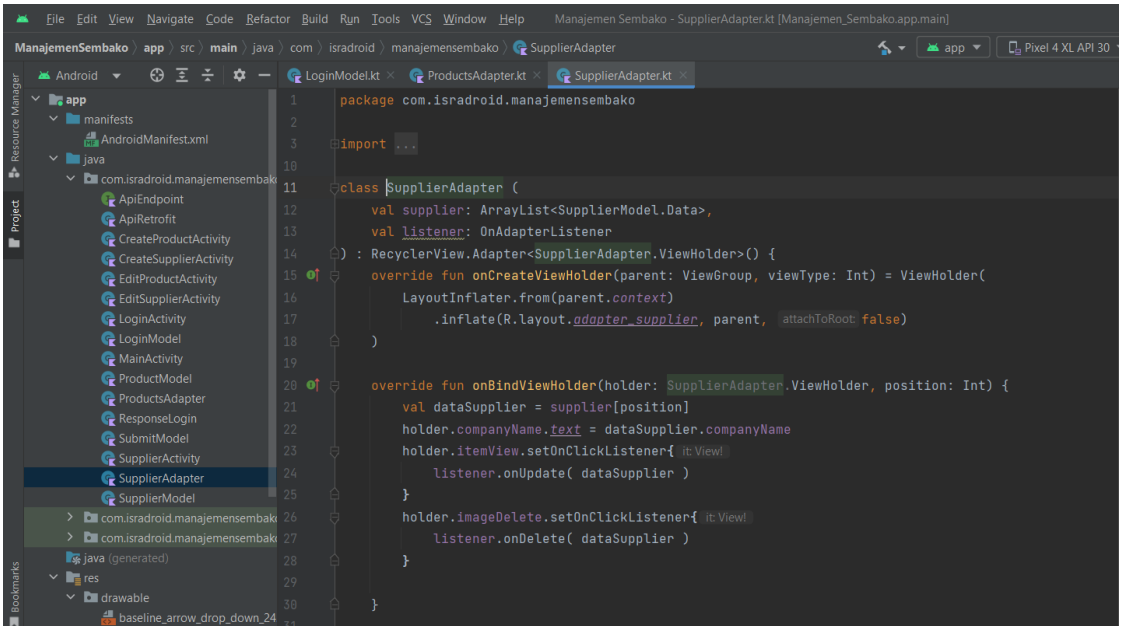
Gambar 2.1.6 Class LoginModel



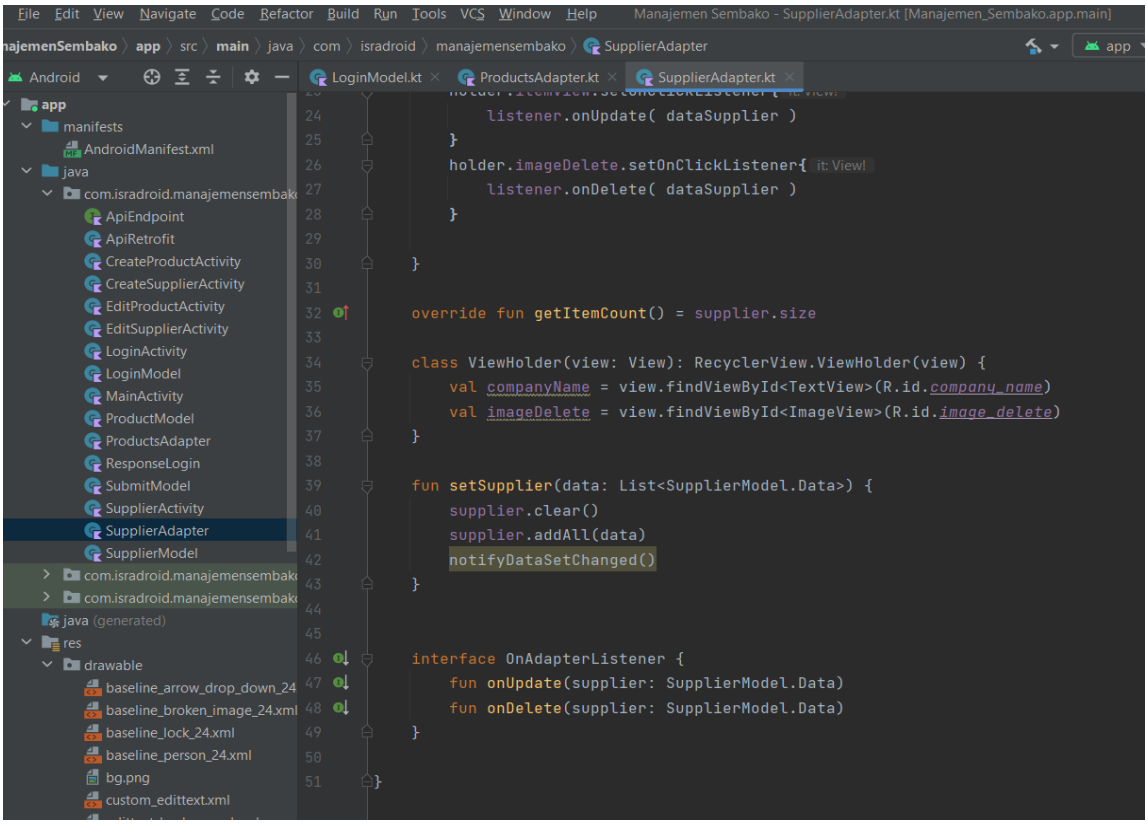
Gambar 2.1.7 Class ProductAdapter



Gambar 2.1.8 ProductAdapter

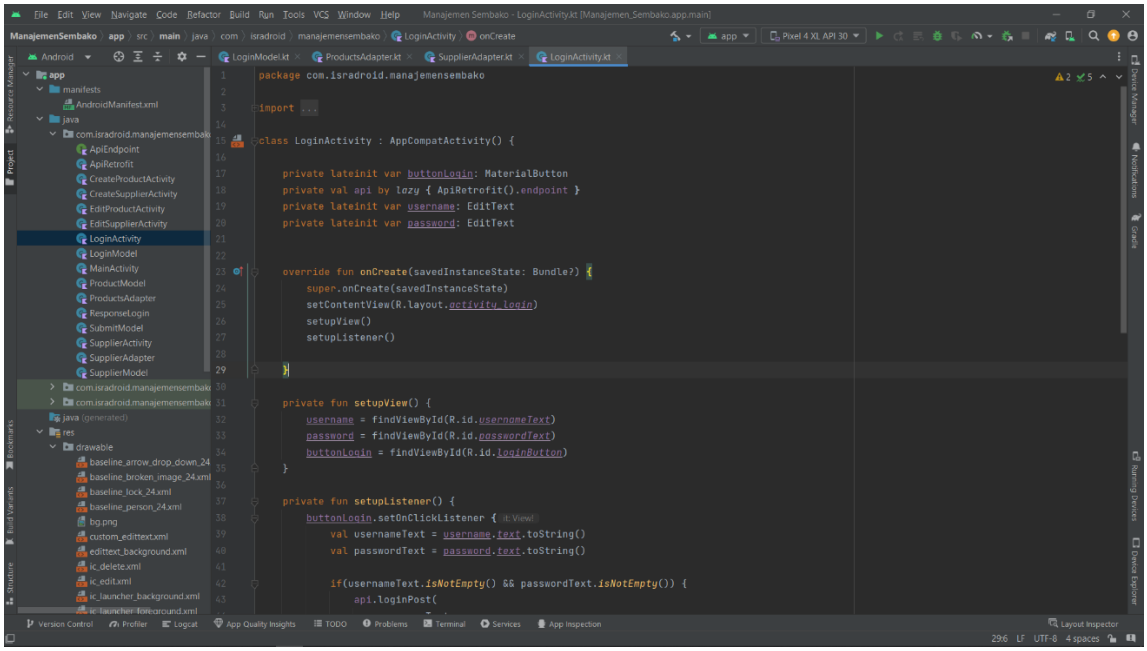


Gambar 2.1.9 SupplierAdapter



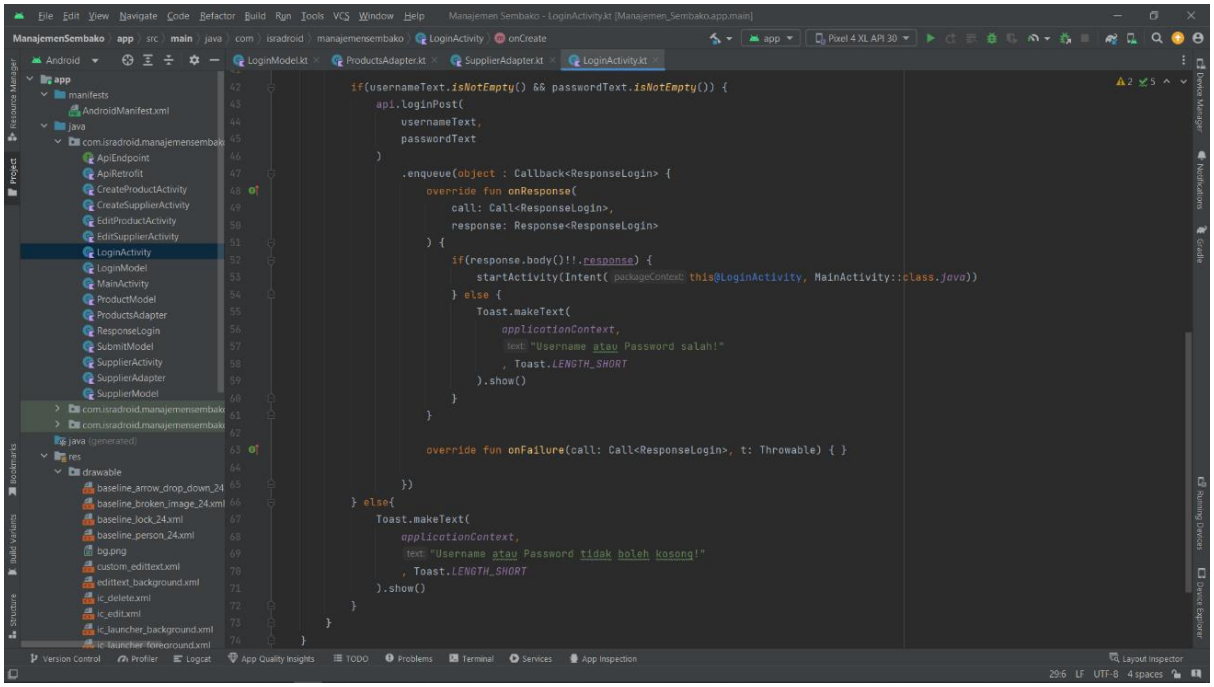
```
            listener.onUpdate( dataSupplier )
        }
        holder.imageDelete.setOnClickListener{ it: View!
            listener.onDelete( dataSupplier )
        }

    }

    override fun getItemCount() = supplier.size

    class ViewHolder(view: View): RecyclerView.ViewHolder(view) {
        val companyName = view.findViewById<TextView>(R.id.company_name)
        val imageDelete = view.findViewById<ImageView>(R.id.image_delete)
    }

    fun setSupplier(data: List<SupplierModel.Data>) {
        supplier.clear()
        supplier.addAll(data)
        notifyDataSetChanged()
    }


    interface OnAdapterListener {
        fun onUpdate(supplier: SupplierModel.Data)
        fun onDelete(supplier: SupplierModel.Data)
    }

}
```

Gambar 2.1.10 SupplierAdapter



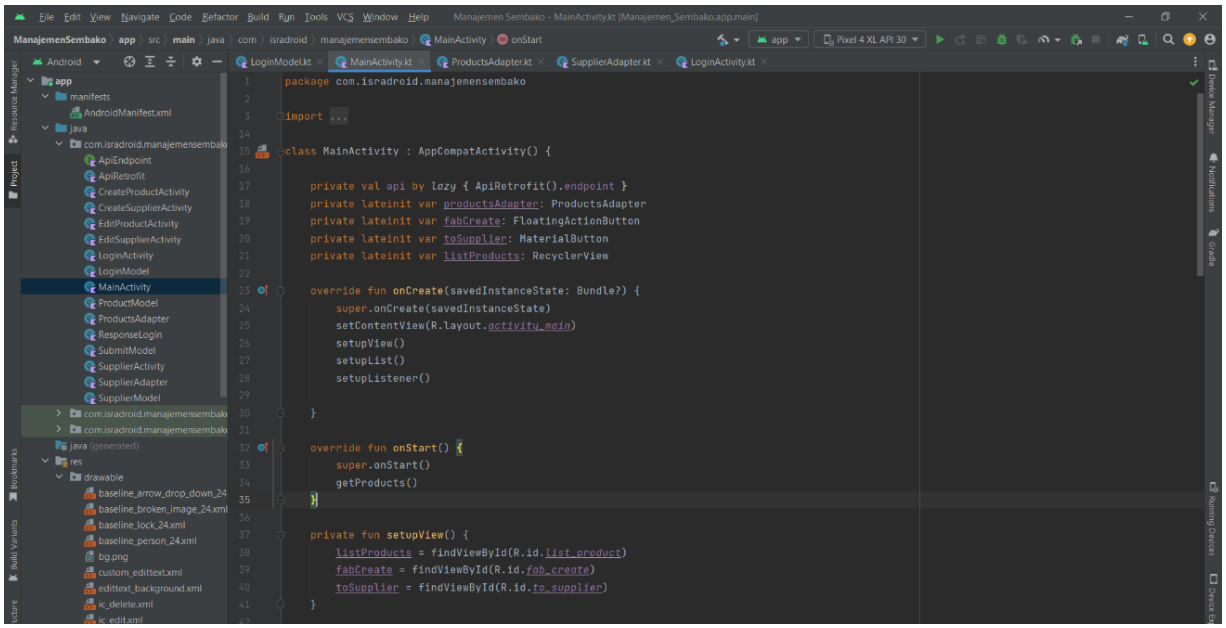
```
package com.isradroid.manajemensembako

import ...

class LoginActivity : AppCompatActivity() {

    private lateinit var buttonLogin: MaterialButton
    private val api by lazy { ApiRetrofit().endpoint }
    private lateinit var username: EditText
    private lateinit var password: EditText


    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_login)
        setupView()
        setupListener()

    }

    private fun setupView() {
        username = findViewById(R.id.usernameText)
        password = findViewById(R.id.passwordText)
        buttonLogin = findViewById(R.id.loginButton)
    }

    private fun setupListener() {
        buttonLogin.setOnClickListener { it: View!
            val usernameText = username.text.toString()
            val passwordText = password.text.toString()

            if(usernameText.isNotEmpty() && passwordText.isNotEmpty()) {
                api.loginPost(
```

Gambar 2.1.11 LoginActivity



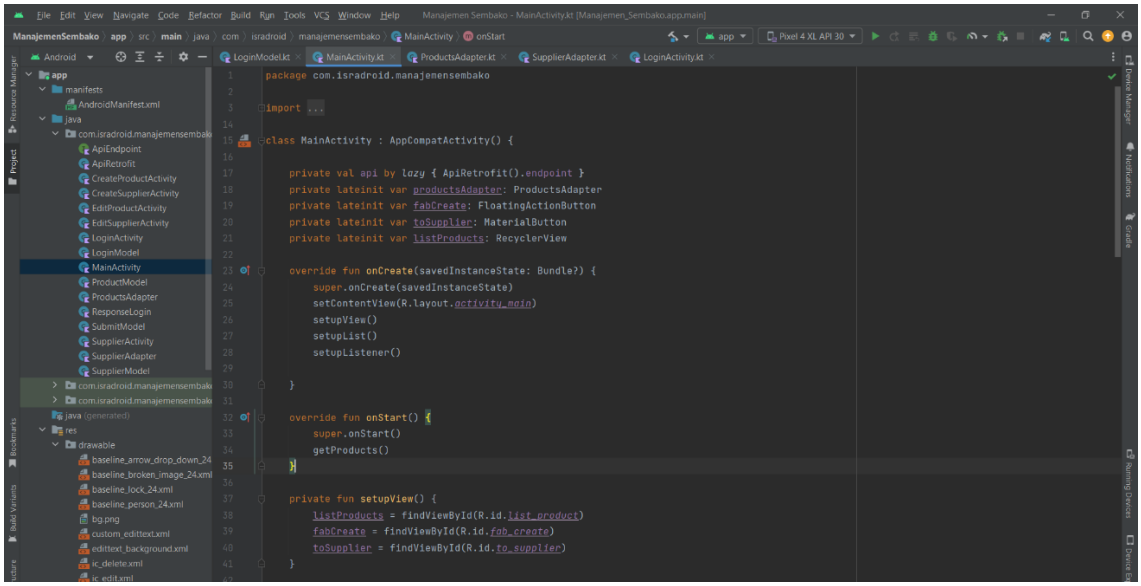
```
if(usernameText.isNotEmpty() && passwordText.isNotEmpty()) {
    api.loginPost(
        usernameText,
        passwordText
    )
        .enqueue(object : Callback<ResponseLogin> {
            override fun onResponse(
                call: Call<ResponseLogin>,
                response: Response<ResponseLogin>
            ) {
                if(response.body()!!.response) {
                    startActivity(Intent( packageContext: this@LoginActivity, MainActivity::class.java))
                } else {
                    Toast.makeText(
                        applicationContext,
                        text: "Username atau Password salah!"
                        , Toast.LENGTH_SHORT
                    ).show()
                }
            }

            override fun onFailure(call: Call<ResponseLogin>, t: Throwable) { }

        })
} else{
    Toast.makeText(
        applicationContext,
        text: "Username atau Password tidak boleh kosong!"
        , Toast.LENGTH_SHORT
    ).show()
}
}
}
```

Gambar 2.1.12 LoginActivity



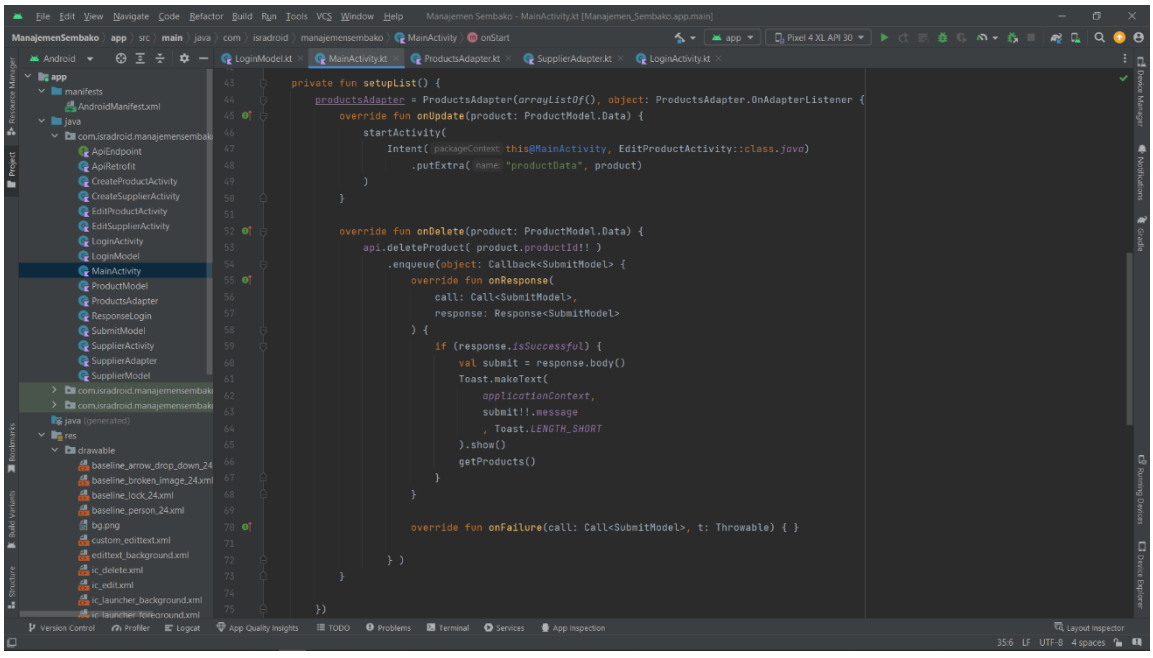
```
package com.isradroid.manajemensembako

import ...

class MainActivity : AppCompatActivity() {

    private val api by lazy { ApiRetrofit().endpoint }
    private lateinit var productsAdapter: ProductsAdapter
    private lateinit var fabCreate: FloatingActionButton
    private lateinit var toSupplier: MaterialButton
    private lateinit var listProducts: RecyclerView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        setupView()
        setupList()
        setupListener()

    }

    override fun onStart() {
        super.onStart()
        getProducts()
    }

    private fun setupView() {
        listProducts = findViewById(R.id.list_product)
        fabCreate = findViewById(R.id.fab_create)
        toSupplier = findViewById(R.id.to_supplier)
    }
```

Gambar 2.1.13 MainActivity



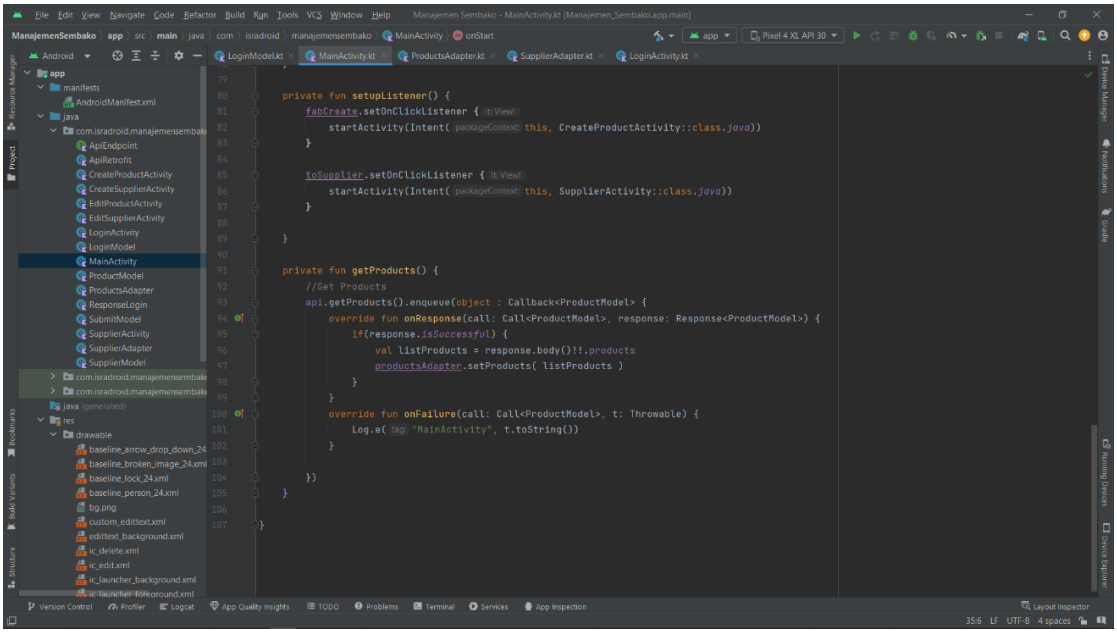
```
package com.isradroid.manajemensembako

import ...

class MainActivity : AppCompatActivity() {

    private val api by lazy { ApiRetrofit().endpoint }
    private lateinit var productsAdapter: ProductsAdapter
    private lateinit var fabCreate: FloatingActionButton
    private lateinit var toSupplier: MaterialButton
    private lateinit var listProducts: RecyclerView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        setupView()
        setupList()
        setupListener()

    }

    override fun onStart() {
        super.onStart()
        getProducts()
    }

    private fun setupView() {
        listProducts = findViewById(R.id.list_product)
        fabCreate = findViewById(R.id.fab_create)
        toSupplier = findViewById(R.id.to_supplier)
    }
```

Gambar 2.1.14 MainActivity



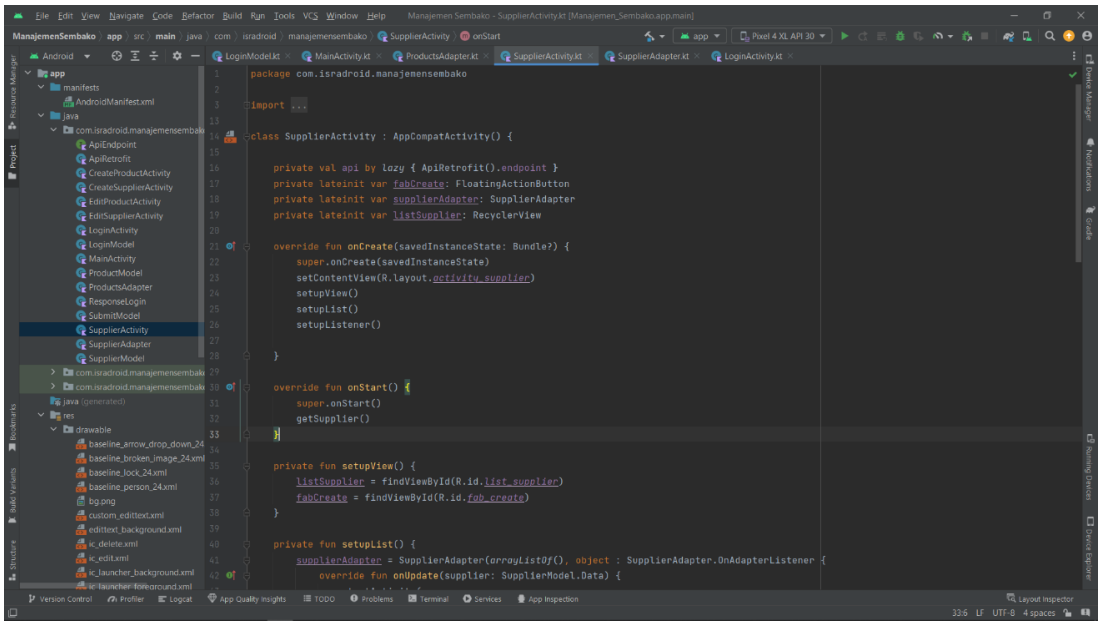
```
private fun setupList() {
    productsAdapter = ProductsAdapter(arrayListOf(), object: ProductsAdapter.OnAdapterListener {
        override fun onUpdate(product: ProductModel.Data) {
            startActivity(
                Intent( packageContext: this@MainActivity, EditProductActivity::class.java)
                    .putExtra( name: "productData", product)
            )
        }

        override fun onDelete(product: ProductModel.Data) {
            api.deleteProduct( product.productId!! )
                .enqueue(object: Callback<SubmitModel> {
                    override fun onResponse(
                        call: Call<SubmitModel>,
                        response: Response<SubmitModel>
                    ) {
                        if (response.isSuccessful) {
                            val submit = response.body()
                            Toast.makeText(
                                applicationContext,
                                submit!!.message
                                , Toast.LENGTH_SHORT
                            ).show()
                            getProducts()
                        }
                    }

                    override fun onFailure(call: Call<SubmitModel>, t: Throwable) { }

                } )
        }
    })
```

Gambar 2.1.15 MainActivity



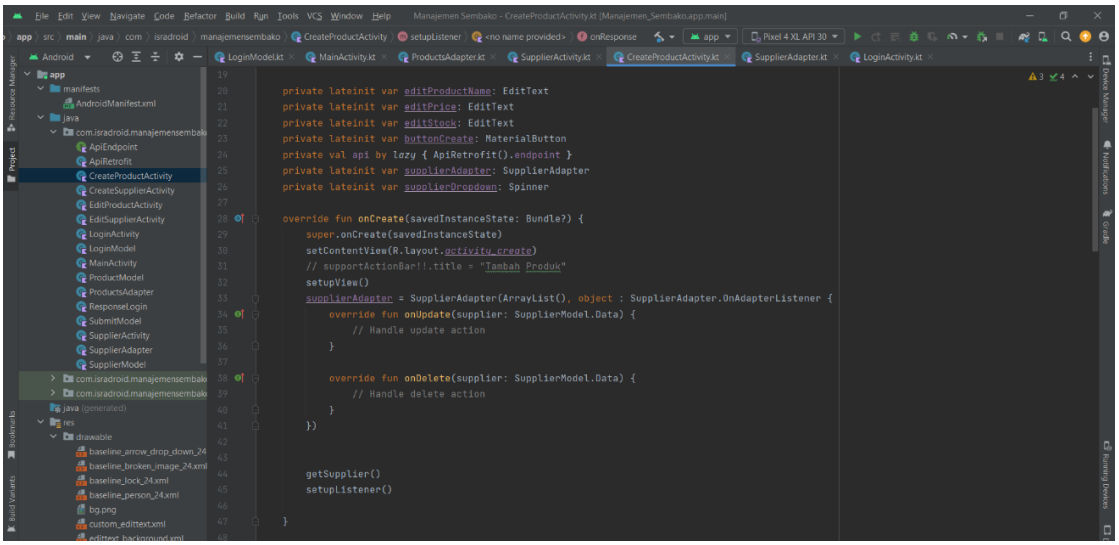
```
    private fun setupListener() {
        fabCreate.setOnClickListener { it:View!
            startActivity(Intent( packageContext: this, CreateProductActivity::class.java))
        }

        toSupplier.setOnClickListener { it:View!
            startActivity(Intent( packageContext: this, SupplierActivity::class.java))
        }

    }

    private fun getProducts() {
        //Get Products
        api.getProducts().enqueue(object : Callback<ProductModel> {
            override fun onResponse(call: Call<ProductModel>, response: Response<ProductModel>) {
                if(response.isSuccessful) {
                    val listProducts = response.body()!!.products
                    productsAdapter.setProducts( listProducts )
                }
            }
            override fun onFailure(call: Call<ProductModel>, t: Throwable) {
                Log.e( tag: "MainActivity", t.toString())
            }

        })
    }

}
```

Gambar 2.1.16 MainActivity



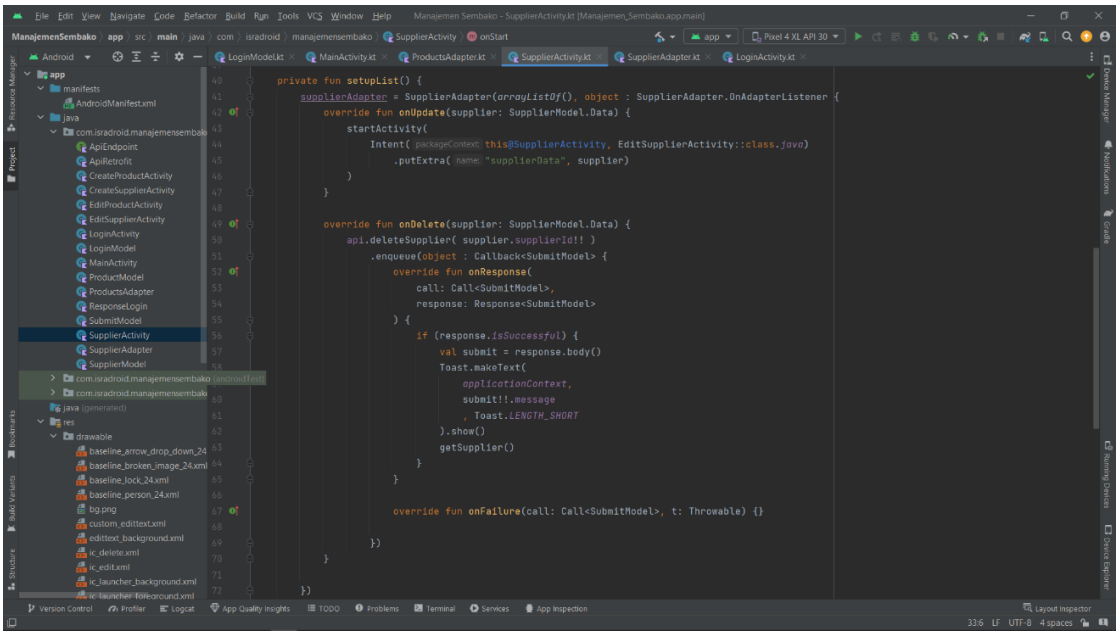
```
package com.isradroid.manajemensembako

import ...

class SupplierActivity : AppCompatActivity() {

    private val api by lazy { ApiRetrofit().endpoint }
    private lateinit var fabCreate: FloatingActionButton
    private lateinit var supplierAdapter: SupplierAdapter
    private lateinit var listSupplier: RecyclerView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_supplier)
        setupView()
        setupList()
        setupListener()

    }

    override fun onStart() {
        super.onStart()
        getSupplier()
    }

    private fun setupView() {
        listSupplier = findViewById(R.id.list_supplier)
        fabCreate = findViewById(R.id.fab_create)
    }

    private fun setupList() {
        supplierAdapter = SupplierAdapter(arrayListOf(), object : SupplierAdapter.OnAdapterListener {
            override fun onUpdate(supplier: SupplierModel.Data) {
```

Gambar 2.1.17 SupplierActivity



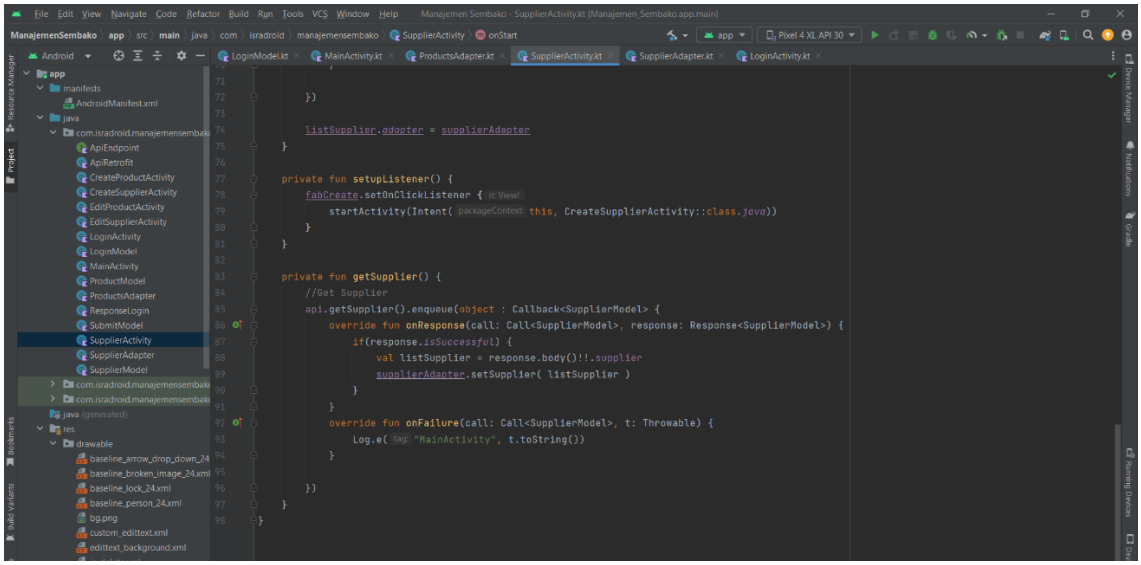
```
    private lateinit var editProductName: EditText
    private lateinit var editPrice: EditText
    private lateinit var editStock: EditText
    private lateinit var buttonCreate: MaterialButton
    private val api by lazy { ApiRetrofit().endpoint }
    private lateinit var supplierAdapter: SupplierAdapter
    private lateinit var supplierDropdown: Spinner

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_create)
        // supportActionBar!!.title = "Tambah Produk"
        setupView()
        supplierAdapter = SupplierAdapter(ArrayList(), object : SupplierAdapter.OnAdapterListener {
            override fun onUpdate(supplier: SupplierModel.Data) {
                // Handle update action
            }

            override fun onDelete(supplier: SupplierModel.Data) {
                // Handle delete action
            }
        })


        getSupplier()
        setupListener()

    }
```

Gambar 2.1.18 CreatProductActivity



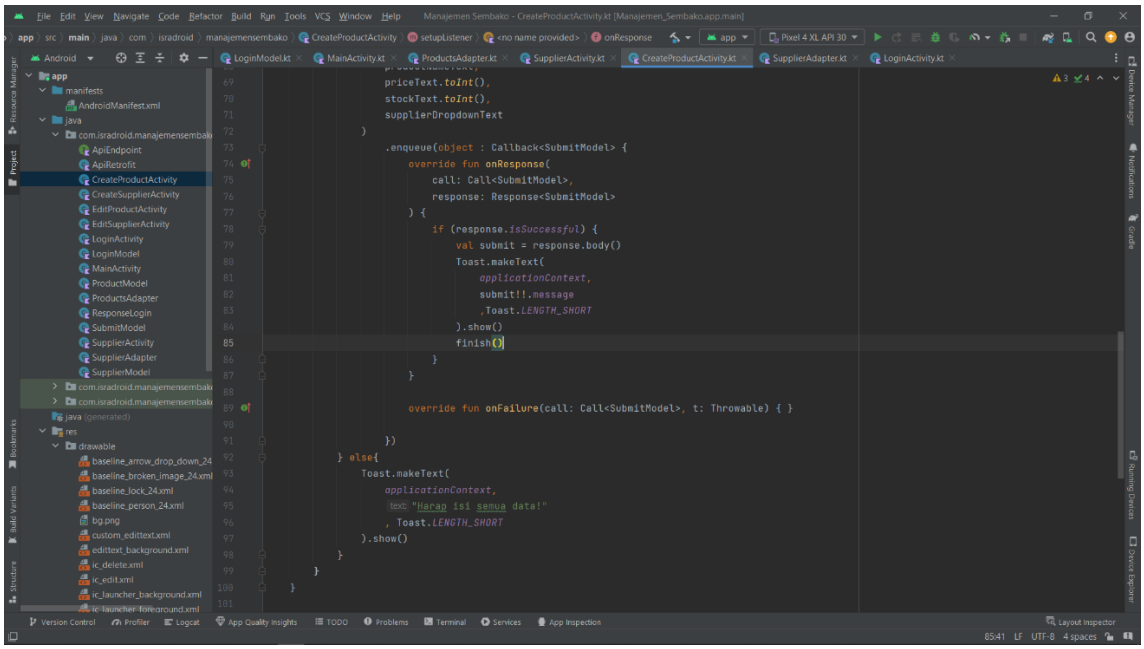
```
    private fun setupList() {
        supplierAdapter = SupplierAdapter(arrayListOf(), object : SupplierAdapter.OnAdapterListener {
            override fun onUpdate(supplier: SupplierModel.Data) {
                startActivity(
                    Intent( packageContext: this@SupplierActivity, EditSupplierActivity::class.java)
                        .putExtra( name: "supplierData", supplier)
                )
            }

            override fun onDelete(supplier: SupplierModel.Data) {
                api.deleteSupplier( supplier.supplierId!! )
                    .enqueue(object : Callback<SubmitModel> {
                        override fun onResponse(
                            call: Call<SubmitModel>,
                            response: Response<SubmitModel>
                        ) {
                            if (response.isSuccessful) {
                                val submit = response.body()
                                Toast.makeText(
                                    applicationContext,
                                    submit!!.message
                                    , Toast.LENGTH_SHORT
                                ).show()
                                getSupplier()
                            }
                        }

                        override fun onFailure(call: Call<SubmitModel>, t: Throwable) {}

                    })
            }
        })
```

Gambar 2.1.19 SupplierActivity



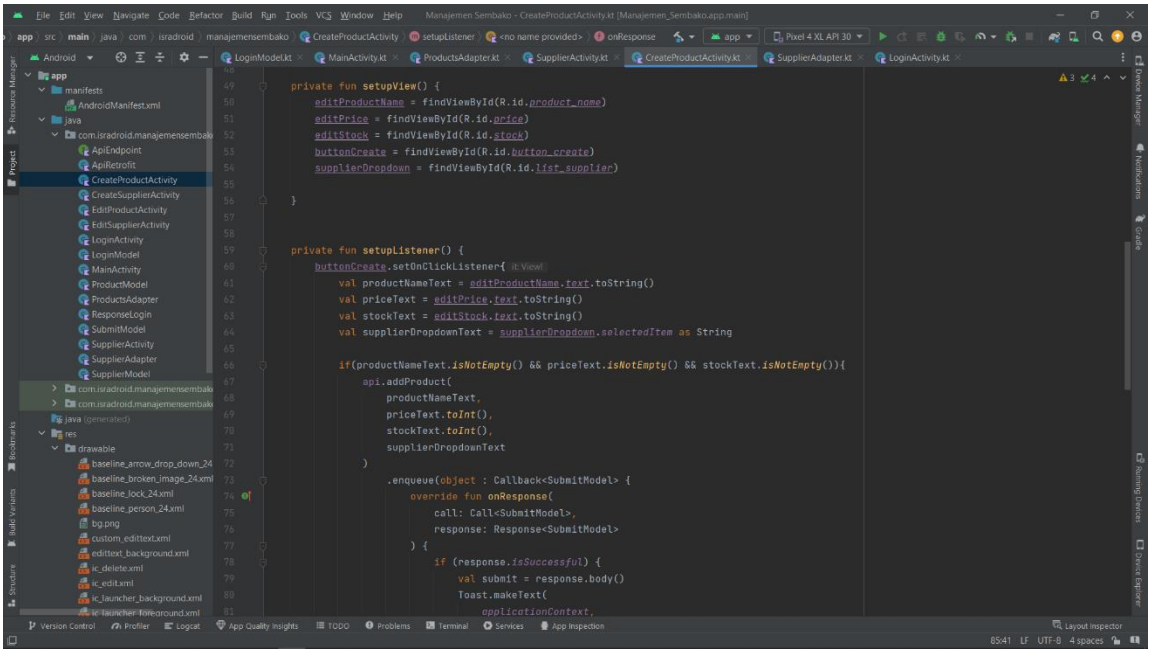
```
        })

        listSupplier.adapter = supplierAdapter
    }

    private fun setupListener() {
        fabCreate.setOnClickListener { it: View!
            startActivity(Intent( packageContext: this, CreateSupplierActivity::class.java))
        }
    }

    private fun getSupplier() {
        //Get Supplier
        api.getSupplier().enqueue(object : Callback<SupplierModel> {
            override fun onResponse(call: Call<SupplierModel>, response: Response<SupplierModel>) {
                if(response.isSuccessful) {
                    val listSupplier = response.body()!!.supplier
                    supplierAdapter.setSupplier( listSupplier )
                }
            }
            override fun onFailure(call: Call<SupplierModel>, t: Throwable) {
                Log.e( tag: "MainActivity", t.toString())
            }

        })
    }
}
```

Gambar 2.1.20 SupplierActivity



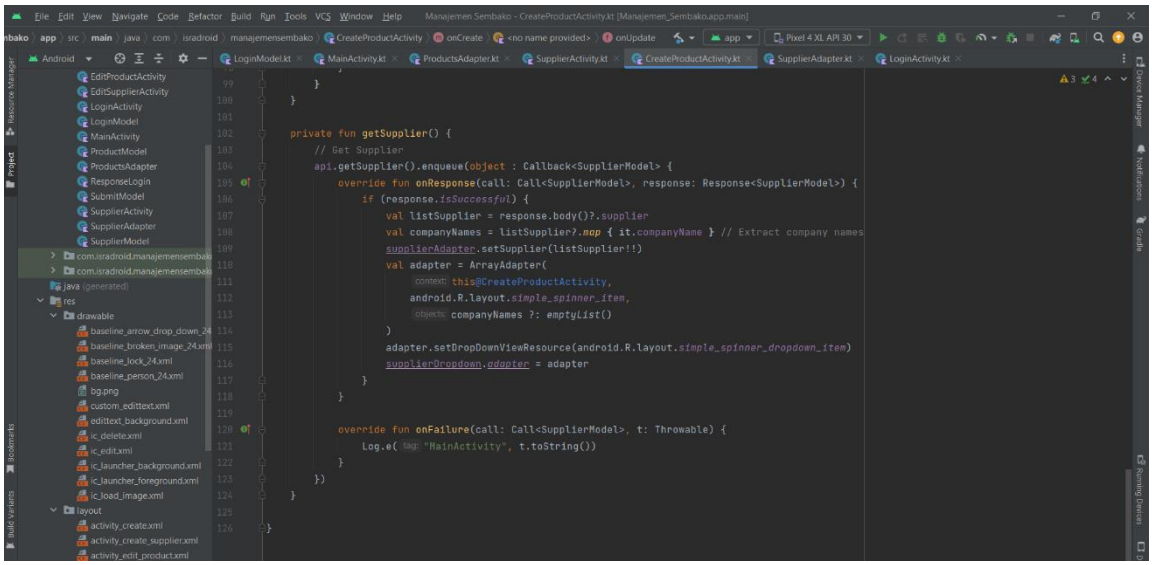
```
                    priceText.toInt(),
                    stockText.toInt(),
                    supplierDropdownText
                )
                    .enqueue(object : Callback<SubmitModel> {
                        override fun onResponse(
                            call: Call<SubmitModel>,
                            response: Response<SubmitModel>
                        ) {
                            if (response.isSuccessful) {
                                val submit = response.body()
                                Toast.makeText(
                                    applicationContext,
                                    submit!!.message
                                    ,Toast.LENGTH_SHORT
                                ).show()
                                finish()
                            }
                        }

                        override fun onFailure(call: Call<SubmitModel>, t: Throwable) { }

                    })
            } else{
                Toast.makeText(
                    applicationContext,
                    text: "Harap isi semua data!"
                    , Toast.LENGTH_SHORT
                ).show()
            }
        }
    }
```

Gambar 2.1.21 CreatPruductActivity



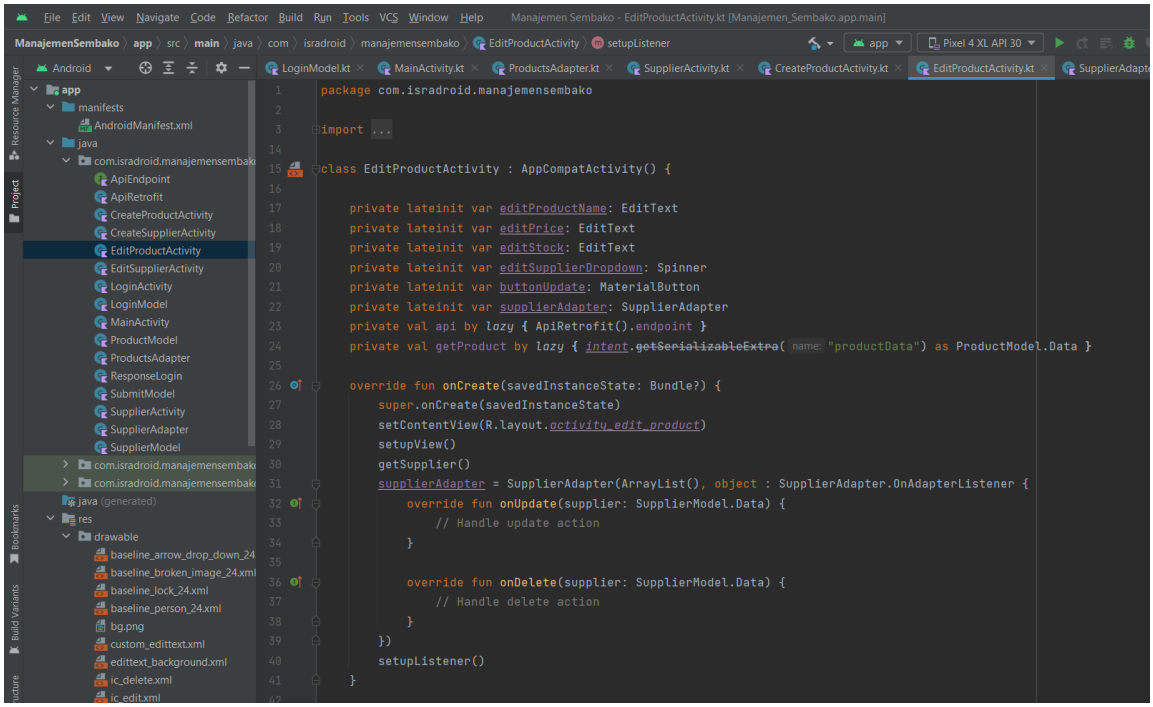
```
private fun setupView() {
    editProductName = findViewById(R.id.product_name)
    editPrice = findViewById(R.id.price)
    editStock = findViewById(R.id.stock)
    buttonCreate = findViewById(R.id.button_create)
    supplierDropdown = findViewById(R.id.list_supplier)

}


private fun setupListener() {
    buttonCreate.setOnClickListener{ it: View!
        val productNameText = editProductName.text.toString()
        val priceText = editPrice.text.toString()
        val stockText = editStock.text.toString()
        val supplierDropdownText = supplierDropdown.selectedItem as String

        if(productNameText.isNotEmpty() && priceText.isNotEmpty() && stockText.isNotEmpty()){
            api.addProduct(
                productNameText,
                priceText.toInt(),
                stockText.toInt(),
                supplierDropdownText
            )
                .enqueue(object : Callback<SubmitModel> {
                    override fun onResponse(
                        call: Call<SubmitModel>,
                        response: Response<SubmitModel>
                    ) {
                        if (response.isSuccessful) {
                            val submit = response.body()
                            Toast.makeText(
                                applicationContext,
```

Gambar 2.1.22 CreateProductActivity



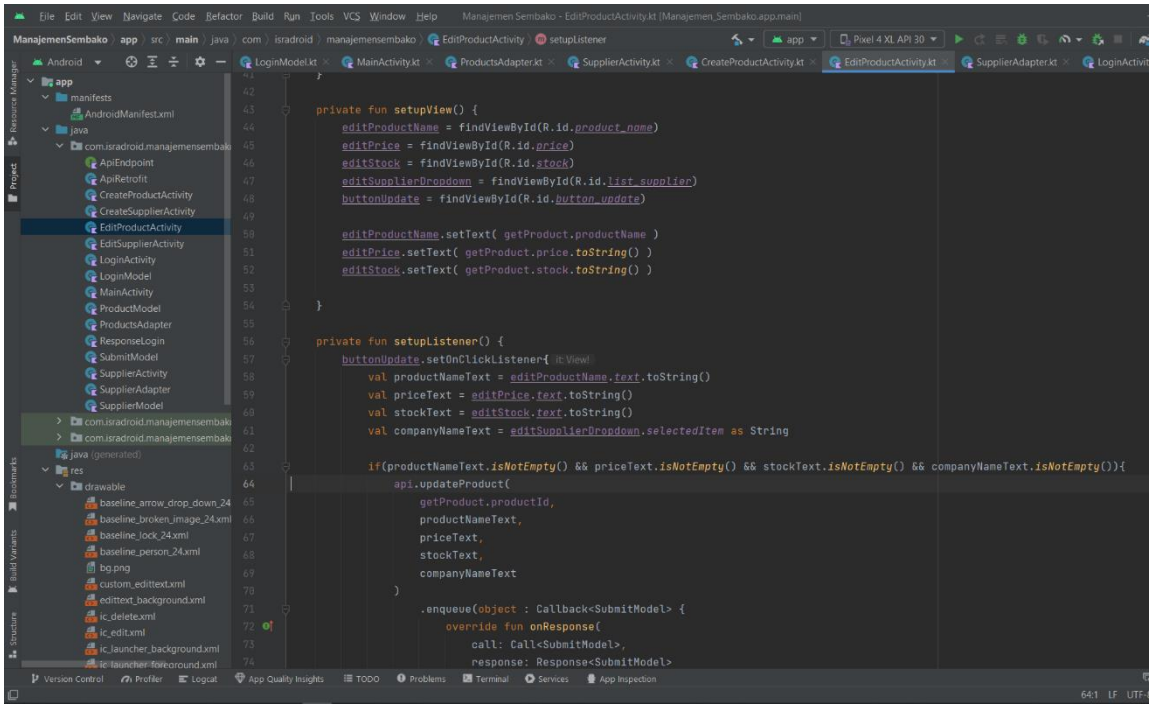
```
        }
    }

    private fun getSupplier() {
        // Get Supplier
        api.getSupplier().enqueue(object : Callback<SupplierModel> {
            override fun onResponse(call: Call<SupplierModel>, response: Response<SupplierModel>) {
                if (response.isSuccessful) {
                    val listSupplier = response.body()?.supplier
                    val companyNames = listSupplier?.map { it.companyName } // Extract company names
                    supplierAdapter.setSupplier(listSupplier!!)
                    val adapter = ArrayAdapter(
                        context: this@CreateProductActivity,
                        android.R.layout.simple_spinner_item,
                        objects: companyNames ?: emptyList()
                    )
                    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item)
                    supplierDropdown.adapter = adapter
                }
            }

            override fun onFailure(call: Call<SupplierModel>, t: Throwable) {
                Log.e( tag: "MainActivity", t.toString())
            }
        })
    }

}
```

Gambar 2.1.23 CreateProductActivity



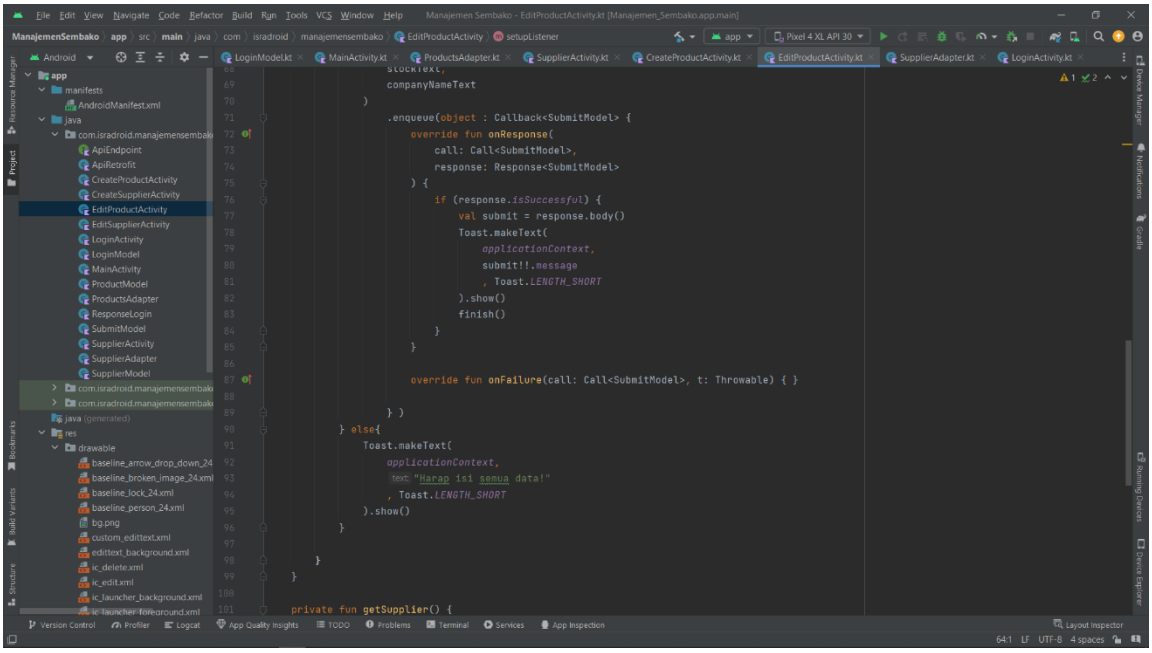
```
package com.isradroid.manajemensembako

import ...

class EditProductActivity : AppCompatActivity() {

    private lateinit var editProductName: EditText
    private lateinit var editPrice: EditText
    private lateinit var editStock: EditText
    private lateinit var editSupplierDropdown: Spinner
    private lateinit var buttonUpdate: MaterialButton
    private lateinit var supplierAdapter: SupplierAdapter
    private val api by lazy { ApiRetrofit().endpoint }
    private val getProduct by lazy { intent.getSerializableExtra( name: "productData") as ProductModel.Data }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_edit_product)
        setupView()
        getSupplier()
        supplierAdapter = SupplierAdapter(ArrayList(), object : SupplierAdapter.OnAdapterListener {
            override fun onUpdate(supplier: SupplierModel.Data) {
                // Handle update action
            }

            override fun onDelete(supplier: SupplierModel.Data) {
                // Handle delete action
            }
        })
        setupListener()
    }
```

Gambar 2.1.24 EditProductActivity

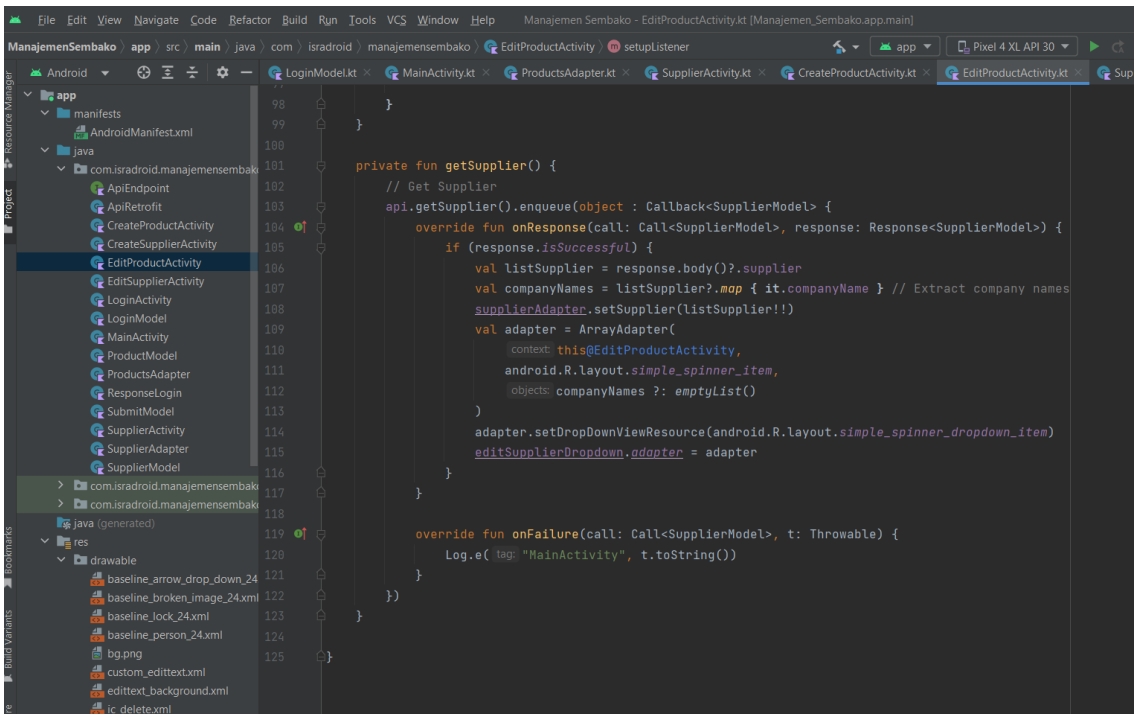


```
    private fun setupView() {
        editProductName = findViewById(R.id.product_name)
        editPrice = findViewById(R.id.price)
        editStock = findViewById(R.id.stock)
        editSupplierDropdown = findViewById(R.id.list_supplier)
        buttonUpdate = findViewById(R.id.button_update)

        editProductName.setText( getProduct.productName )
        editPrice.setText( getProduct.price.toString() )
        editStock.setText( getProduct.stock.toString() )

    }

    private fun setupListener() {
        buttonUpdate.setOnClickListener{ it: View!
            val productNameText = editProductName.text.toString()
            val priceText = editPrice.text.toString()
            val stockText = editStock.text.toString()
            val companyNameText = editSupplierDropdown.selectedItem as String

            if(productNameText.isNotEmpty() && priceText.isNotEmpty() && stockText.isNotEmpty() && companyNameText.isNotEmpty()){
                api.updateProduct(
                    getProduct.productId,
                    productNameText,
                    priceText,
                    stockText,
                    companyNameText
                )
                    .enqueue(object : Callback<SubmitModel> {
                        override fun onResponse(
                            call: Call<SubmitModel>,
                            response: Response<SubmitModel>
```

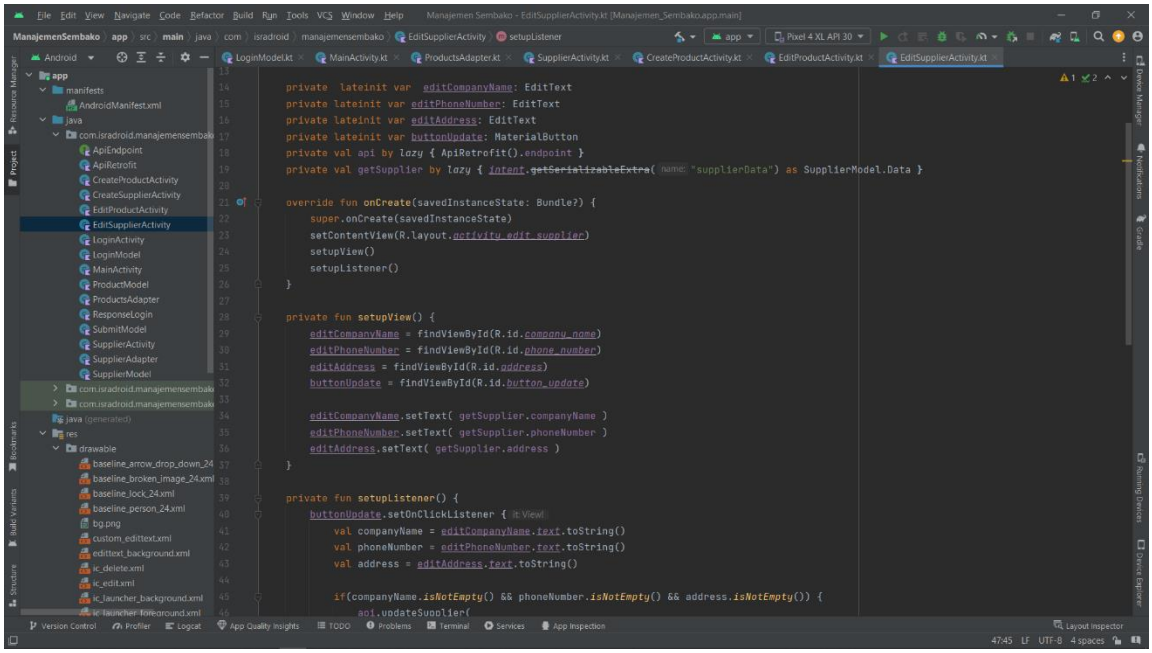
Gambar 2.1.25 EditProductActivity



Gambar 2.1.25 EditProductActivity

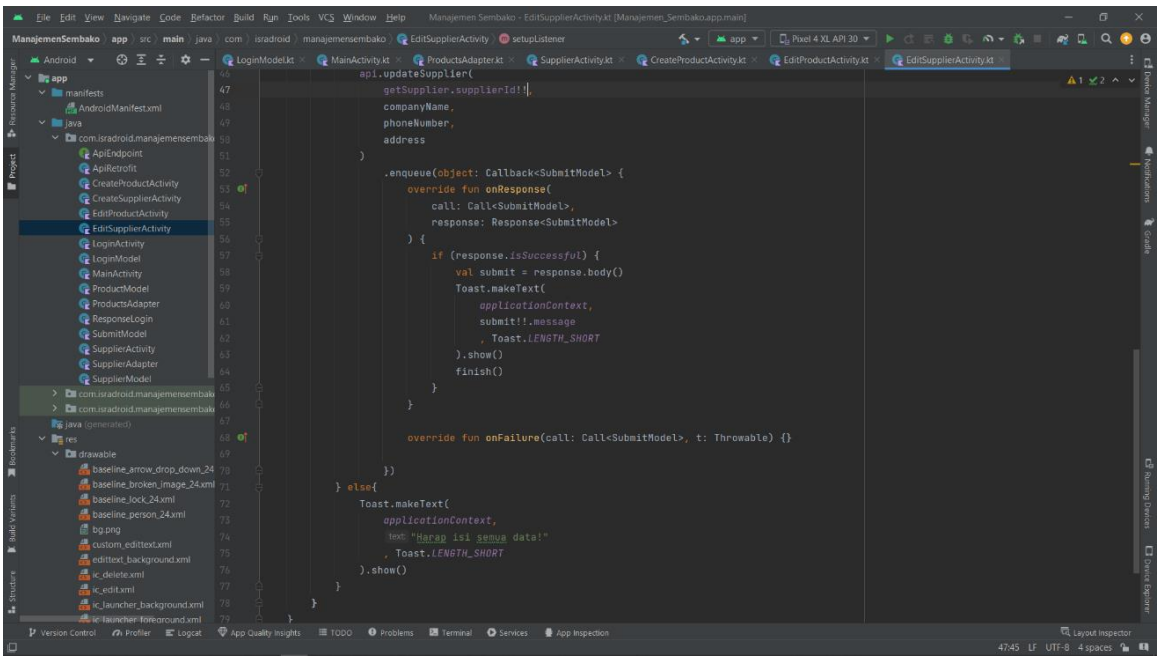


Gambar 2.1.26 EditProductActivity



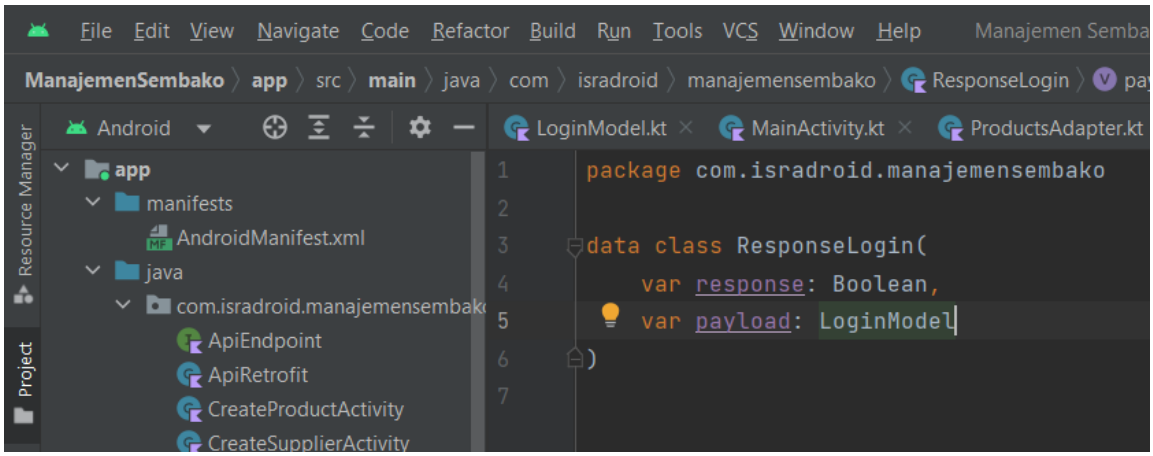
```
    private  lateinit var  editCompanyName: EditText
    private lateinit var editPhoneNumber: EditText
    private lateinit var editAddress: EditText
    private lateinit var buttonUpdate: MaterialButton
    private val api by lazy { ApiRetrofit().endpoint }
    private val getSupplier by lazy { intent.getSerializableExtra( name: "supplierData") as SupplierModel.Data }

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_edit_supplier)
        setupView()
        setupListener()
    }

    private fun setupView() {
        editCompanyName = findViewById(R.id.company_name)
        editPhoneNumber = findViewById(R.id.phone_number)
        editAddress = findViewById(R.id.address)
        buttonUpdate = findViewById(R.id.button_update)

        editCompanyName.setText( getSupplier.companyName )
        editPhoneNumber.setText( getSupplier.phoneNumber )
        editAddress.setText( getSupplier.address )
    }

    private fun setupListener() {
        buttonUpdate.setOnClickListener { it:View!
            val companyName = editCompanyName.text.toString()
            val phoneNumber = editPhoneNumber.text.toString()
            val address = editAddress.text.toString()

            if(companyName.isNotEmpty() && phoneNumber.isNotEmpty() && address.isNotEmpty()) {
                api.updateSupplier(
```

Gambar 2.1.27 EditSupplierActivity

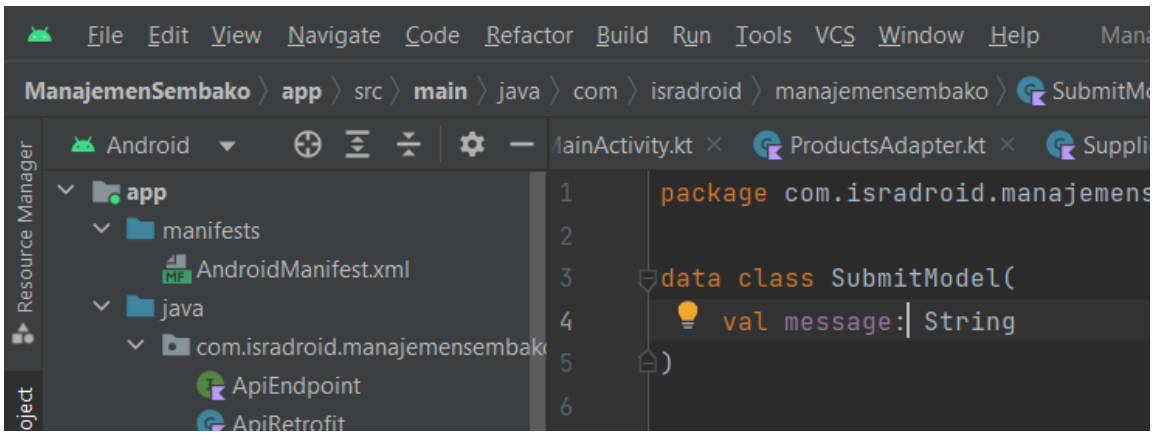


```
                api.updateSupplier(
                    getSupplier.supplierId!!,
                    companyName,
                    phoneNumber,
                    address
                )
                    .enqueue(object: Callback<SubmitModel> {
                        override fun onResponse(
                            call: Call<SubmitModel>,
                            response: Response<SubmitModel>
                        ) {
                            if (response.isSuccessful) {
                                val submit = response.body()
                                Toast.makeText(
                                    applicationContext,
                                    submit!!.message
                                    , Toast.LENGTH_SHORT
                                ).show()
                                finish()
                            }
                        }

                        override fun onFailure(call: Call<SubmitModel>, t: Throwable) {}

                    })
            } else{
                Toast.makeText(
                    applicationContext,
                    text: "Harap isi semua data!"
                    , Toast.LENGTH_SHORT
                ).show()
            }
        }
    }
}
```

Gambar 2.1.28 EditSupplierActivity



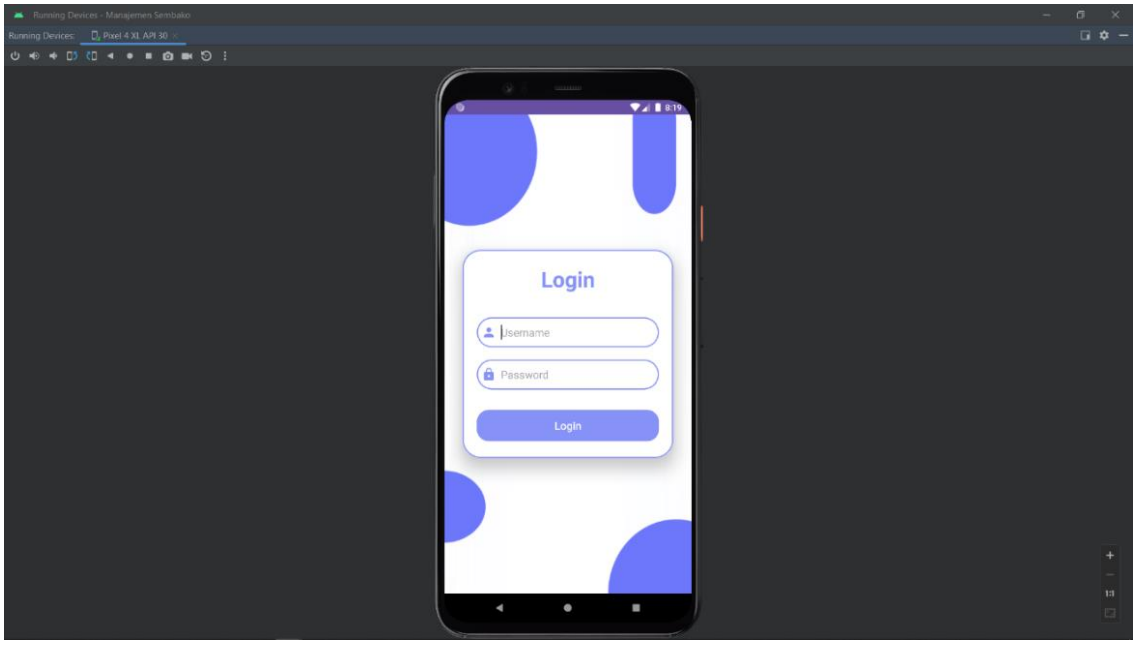
Gambar 2.1.29 Class ResponseLogin



Gambar 2.1.30 Class SubmitModel

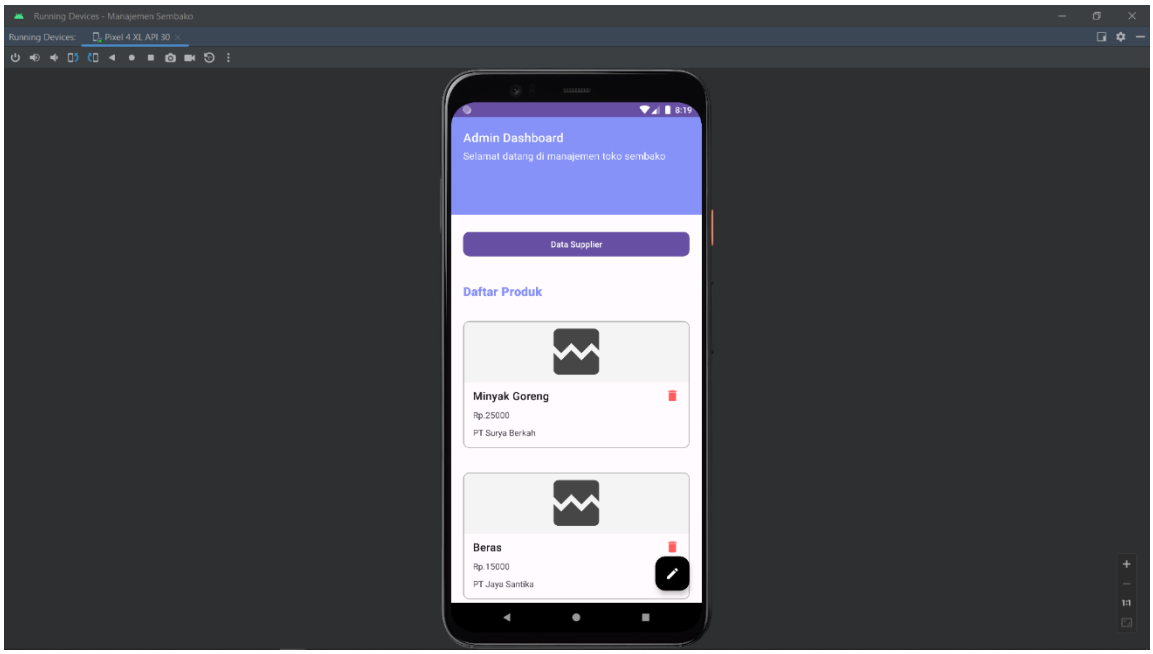
2.2. Cara Kerja Aplikasi

Pada activity ini user diminta untuk memasukan username dan password yang terdaftar di database



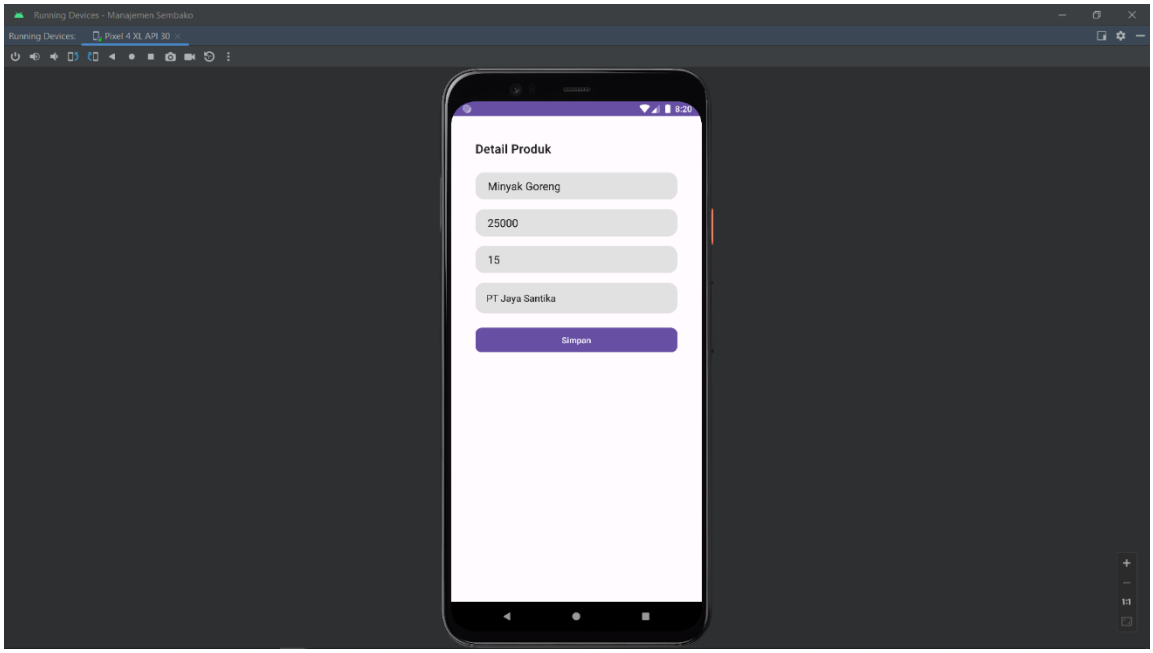
Gambar 2.2.1 Tampilan Login

Kemudian pada activity ini kita dapat menampilkan daftar produk dan terdapat button untuk beralih ke activity supplier serta floating button untuk beralih ke activity tambah produk



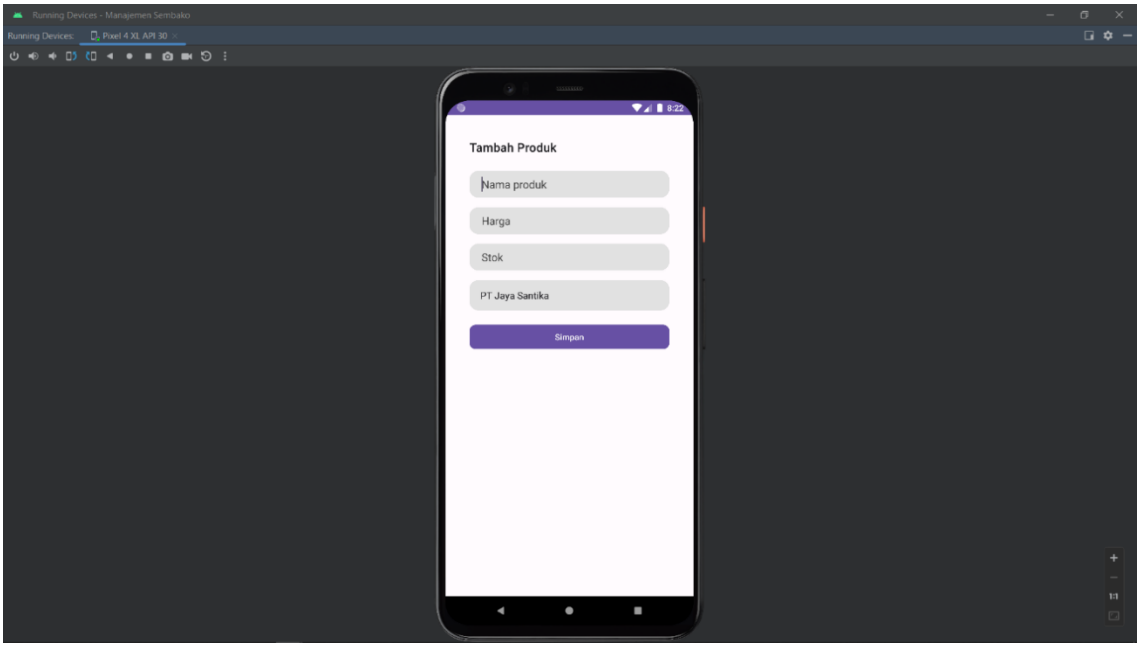
Gambar 2.2.2 Tampilan Dashboard

Di activity ini dapat menampilkan detail produk sekaligus dapat meng-update data dari produk tersebut



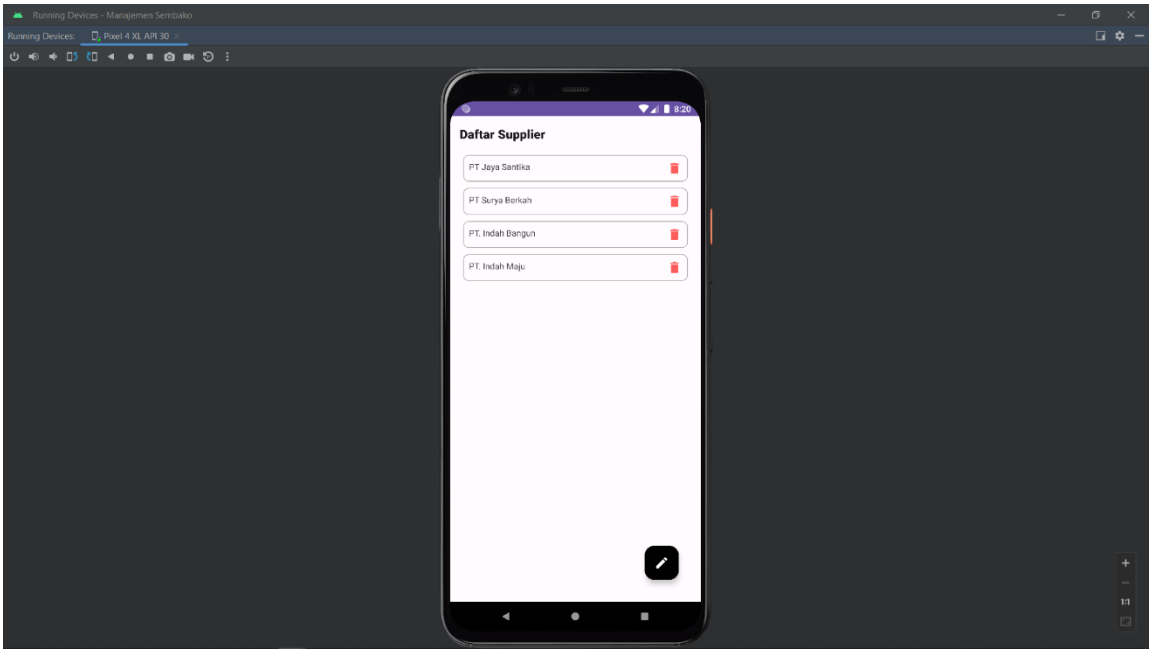
Gambar 2.2.3 Tampilan DetailProduk

Kemudian pada activity ini dapat menambahkan produk baru



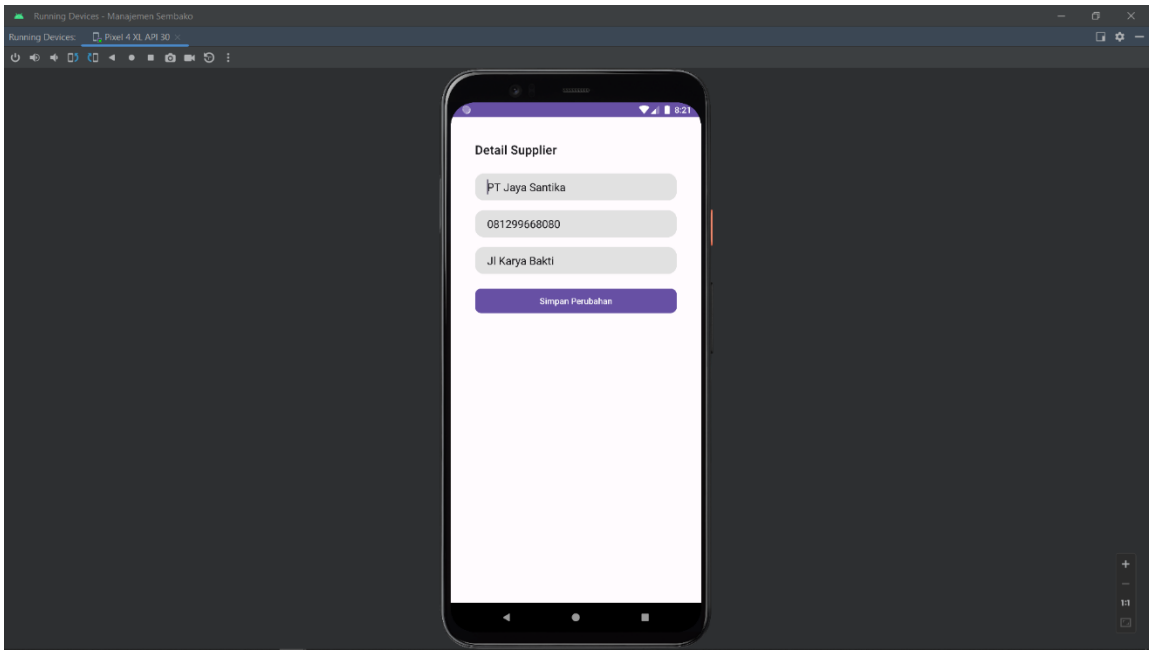
Gambar 2.1.4 Tampilan TambahProduk

Pada activity ini kita dapat menampilkan daftar supplier dan terdapat floating button untuk beralih ke activity tambah supplier



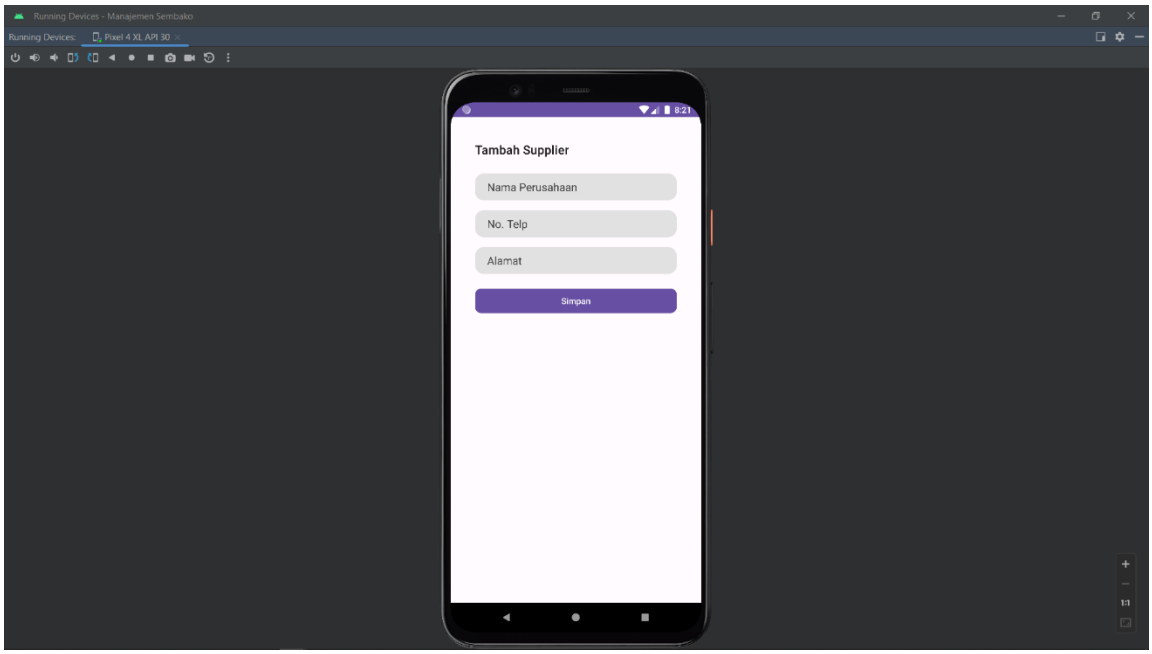
Gambar 2.2.5 Tampilan DataSupplier

Di activity ini dapat menampilkan detail supplier sekaligus dapat meng-update data dari supplier tersebut



Gambar 2.2.7 Tampilan DetailSupplier

Kemudian pada activity ini dapat menambahkan supplier baru .



Gambar 2.2.6 Tampilan TambahSupplier

BAB III PENUTUP

3.1 Simpulan

Berdasarkan pembahasan bab-bab sebelumnya maka pada bab ini penulis dapat mengambil simpulan sebagai berikut:

1. Dengan menggunakan aplikasi ini, pemilik toko dapat meningkatkan efisiensi operasional dengan mengotomatiskan beberapa proses, seperti pencatatan penjualan, manajemen persediaan, dan analisis data.
2. Aplikasi manajemen toko sembako merupakan solusi yang sangat berguna dalam mengoptimalkan operasional toko sembako dan meningkatkan daya saing bisnis.
3. Aplikasi ini juga membantu dalam pengelolaan stok dan persediaan dengan memberikan informasi yang akurat dan real-time.

3.2 Saran

Berdasarkan pembahasan bab-bab sebelumnya maka pada bab ini penulis dapat mengambil simpulan sebagai berikut:

1. Dalam mengembangkan aplikasi manajemen toko sembako, penting untuk mempertimbangkan kebutuhan operasional yang spesifik dari toko sembako. Identifikasi proses bisnis yang perlu diotomatisasi dan fungsionalitas yang diperlukan untuk meningkatkan efisiensi operasional.
2. Pastikan aplikasi manajemen toko sembako memiliki fitur yang memadai untuk mengelola stok dan persediaan. Hal ini mencakup kemampuan untuk mencatat masuk dan keluarnya produk, pemantauan level persediaan, pengingat untuk mengisi ulang persediaan yang menipis, dan kemampuan untuk menghasilkan laporan stok secara berkala.
3. Jaga keamanan data dengan menggunakan sistem keamanan yang handal dalam aplikasi manajemen toko sembako. Ini termasuk proteksi data pelanggan, enkripsi data sensitif, dan pengaturan izin akses yang tepat untuk pengguna aplikasi.
4. Selalu lakukan pembaruan dan perbaikan rutin pada aplikasi untuk memastikan pemeliharaan yang baik dan peningkatan fungsionalitas.