Jelaskan fungsi setiap baris source code pada file kotlin dan submit dalam bentuk pdf pada Elearning

1. SplashScreenActivity.kt

→ Menentukan namespace (nama paket) aplikasi. Digunakan untuk mengorganisir file dan menghindari konflik nama.

```
package com.example.tigapuluhharisehat
```

→ Mengimpor class-class Android yang diperlukan:

- Intent: untuk berpindah ke Activity lain.
- Handler dan Looper: digunakan untuk menunda aksi selama 3 detik.
- AppCompatActivity: class dasar untuk Activity modern di Android.

```
import android.content.Intent
import android.os.Bundle
import android.os.Handler
import android.os.Looper
import androidx.appcompat.app.AppCompatActivity
```

→ Mendefinisikan class `SplashScreenActivity` yang merupakan Activity turunan dari `AppCompatActivity`.

```
class SplashScreenActivity : AppCompatActivity() {
```

→ Membuat variabel untuk menyimpan waktu delay splash screen, yaitu 3000 milidetik (3 detik).

```
private val splashTimeOut: Long = 3000 // 3 detik
```

→ Fungsi onCreate() adalah lifecycle method yang pertama kali dipanggil saat Activity dibuat.

- setContentView(...) menghubungkan layout XML activity_splash_screen.xml ke Activity ini.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_splash_screen)
```

→ Setelah 3 detik, aplikasi akan:

- Membuat `Intent` ke `LoginActivity`
- Menjalankan `LoginActivity` dengan `startActivity`
- Memanggil `finish()` untuk menutup SplashScreen agar tidak bisa dikembalikan dengan tombol "back"

```
Handler(Looper.getMainLooper()).postDelayed({
    val intent = Intent(this, LoginActivity::class.java)
    startActivity(intent)
    finish()
}, splashTimeOut)
```

2. LoginActivity.kt

→ Menyatakan bahwa file ini bagian dari paket `com.example.tigapuluhharisehat`.

```
package com.example.tigapuluhharisehat
```

→ Mengimpor class yang dibutuhkan:

- Intent untuk berpindah activity
- EditText dan Button untuk input dan aksi tombol
- Toast untuk menampilkan pesan singkat
- Patterns untuk validasi email
- AppCompatActivity sebagai superclass Activity

```
import android.content.Intent
import android.os.Bundle
import android.widget.Button
import android.widget.EditText
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import android.util.Patterns
```

→ Membuat class `LoginActivity` yang merupakan turunan dari `AppCompatActivity`.

```
class LoginActivity : AppCompatActivity() {
```

→ Method `onCreate()` dipanggil saat Activity pertama kali dibuat. Layout `activity_login.xml` ditampilkan di layar.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_login) // Pastikan layout ini ada
```

→ Menghubungkan komponen `EditText` dan `Button` dari XML ke variabel Kotlin menggunakan `findViewById`.

```
val edtEmail = findViewById<EditText>(R.id.edtEmail)
val edtPassword = findViewById<EditText>(R.id.edtPassword)
val btnLogin = findViewById<Button>(R.id.btnLogin)
```

→ Menambahkan listener (aksi) saat tombol login ditekan.

```
btnLogin.setOnClickListener {
```

→ Mengambil dan membersihkan input email dan password dari `EditText`.

```
val email = edtEmail.text.toString().trim()
val password = edtPassword.text.toString().trim()
```

→ Validasi: jika email kosong, tampilkan pesan error di EditText dan arahkan fokus ke situ.

```
if (email.isEmpty()) {
    edtEmail.error = getString(R.string.email_empty)
    edtEmail.requestFocus()
```

→ Validasi: jika format email salah, tampilkan error.

```
} else if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
    edtEmail.error = getString(R.string.invalid_email)
    edtEmail.requestFocus()
```

→ Validasi: jika password kosong, tampilkan error.

```
} else if (password.isEmpty()) {
    edtPassword.error = getString(R.string.password_empty)
    edtPassword.requestFocus()
```

→ Jika semua input valid, tampilkan pesan sukses login.

```
} else {
Toast.makeText(this, getString(R.string.login_success),
```

→ Pindah ke halaman `ListChatActivity` dan tutup `LoginActivity` agar tidak bisa diakses dengan tombol back.

3. RegisterActivity.kt

→ Menandakan bahwa file ini berada dalam paket `com.example.tigapuluhharisehat`.

```
package com.example.tigapuluhharisehat
```

→ Mengimpor class Android yang dibutuhkan:

- Intent: navigasi antar activity.
- Patterns: validasi email.
- EditText, Button, ImageView: komponen UI.
- Toast: untuk menampilkan pesan pendek.
- AppCompatActivity: superclass untuk activity modern.

```
import android.content.Intent
import android.os.Bundle
import android.util.Patterns
```

```
import android.widget.Button
import android.widget.EditText
import android.widget.ImageView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
```

→ Mendefinisikan kelas `RegisterActivity` sebagai turunan dari `AppCompatActivity`.

```
class RegisterActivity : AppCompatActivity() {
```

→ `onCreate` dijalankan saat Activity dibuat. Layout `activity_register.xml` ditampilkan sebagai tampilan utama activity ini.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_register)
```

→ Menghubungkan komponen dari XML (EditText, Button, ImageView) dengan kode Kotlin menggunakan `findViewById`.

```
val edtNama = findViewById<EditText>(R.id.edtNama)
val edtEmail = findViewById<EditText>(R.id.edtEmail)
val edtPassword = findViewById<EditText>(R.id.edtPassword)
val btnRegister = findViewById<Button>(R.id.btnRegister)
val imgProfile = findViewById<ImageView>(R.id.imgProfile)
```

→ Memberikan deskripsi pada `ImageView` untuk aksesibilitas (optional tapi bagus untuk UX).

```
imgProfile.contentDescription = "Logo Makanan Sehat"
```

→ Menambahkan event listener saat tombol "Register" ditekan.

```
btnRegister.setOnClickListener {
```

→ Mengambil input dari pengguna, lalu membersihkannya dari spasi kosong di awal/akhir.

```
val nama = edtNama.text.toString().trim()
val email = edtEmail.text.toString().trim()
val password = edtPassword.text.toString().trim()
```

→ Validasi: jika nama kosong, tampilkan pesan error dan arahkan kursor ke kolom tersebut.

```
if (nama.isEmpty()) {
    edtNama.error = getString(R.string.nama_empty)
    edtNama.requestFocus()
```

→ Validasi email: cek apakah email kosong dan apakah format email valid.

```
} else if (email.isEmpty()) {
    edtEmail.error = getString(R.string.email_empty)
    edtEmail.requestFocus()
} else if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
    edtEmail.error = getString(R.string.invalid_email)
    edtEmail.requestFocus()
```

→ Validasi password: jika kosong, tampilkan error.

```
} else if (password.isEmpty()) {
    edtPassword.error = getString(R.string.password_empty)
```

→ Jika semua input valid, tampilkan pesan sukses pendaftaran menggunakan `Toast`.

```
} else {
    // Registrasi berhasil
    Toast.makeText(this, getString(R.string.registration_success),
Toast.LENGTH_SHORT).show()
```

→ Pindah ke `LoginActivity` dan menutup halaman registrasi agar tidak bisa dikembalikan dengan tombol "back".

```
val intent = Intent(this, LoginActivity::class.java)
startActivity(intent)
 finish()  // Menutup RegisterActivity
```

4. ListChatActivity.kt

→ Menandakan file ini berada dalam package utama aplikasi kamu.

```
package com.example.tigapuluhharisehat
```

→ Mengimpor komponen penting:

- `Bundle` & `AppCompatActivity` untuk activity.
- `RecyclerView` & `LinearLayoutManager` untuk menampilkan daftar percakapan.
- `ChatMessage` dari folder `model`.

```
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.example.tigapuluhharisehat.model.ChatMessage
```

→ Mendefinisikan `ListChatActivity` sebagai turunan dari `AppCompatActivity`.

```
class ListChatActivity : AppCompatActivity() {
```

→ `onCreate()` dipanggil saat activity dimulai. Layout yang digunakan adalah `activity_list_chat.xml`.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_list_chat)
```

→ Daftar objek `ChatMessage` berisi nama pengirim, isi pesan, dan waktu. Ini digunakan untuk mengisi tampilan list chat.

```
val chatList = listOf(
    ChatMessage("User", "Apa makanan sehat?", "10:00 AM"),
```

```
    ChatMessage("Ahli Gizi", "Sayur, buah, dan protein tanpa lemak.",
"10:01 AM"),
```

🔁 Menampilkan Chat dengan RecyclerView

→ Menghubungkan RecyclerView dari layout dan mengatur tampilannya dalam bentuk vertikal (seperti daftar chat).

```
val recyclerView: RecyclerView = findViewById(R.id.recyclerViewChat)
recyclerView.layoutManager = LinearLayoutManager(this)
```

→ Membuat adapter `ChatAdapter` dari data `chatList` dan menghubungkannya ke RecyclerView.

```
        val adapter = ChatAdapter(chatList)
        recyclerView.adapter = adapter
    }
}
```

5. ChatAdapter.kt

→ Menandakan file berada di package utama.

```
package com.example.tigapuluhharisehat
```

→ Mengimpor komponen UI dan `ChatMessage`.

```
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.TextView
import android.widget.Toast
import androidx.recyclerview.widget.RecyclerView
import com.example.tigapuluhharisehat.model.ChatMessage
```

Kelas ChatAdapter

→ Kelas ChatAdapter bertanggung jawab untuk menghubungkan data chat ke RecyclerView. Data diambil dari List<ChatMessage>.

```
class ChatAdapter(private val chatList: List<ChatMessage>) :
    RecyclerView.Adapter<ChatAdapter.ChatViewHolder>() {
```

Inner Class ViewHolder

→ Menghubungkan `item_chat.xml` dengan properti TextView.

```
inner class ChatViewHolder(itemView: View) :
RecyclerView.ViewHolder(itemView) {
    val txtSender: TextView = itemView.findViewById(R.id.txtSender)
    val txtMessage: TextView = itemView.findViewById(R.id.txtMessage)
    val txtTime: TextView = itemView.findViewById(R.id.txtTime)
```

→ Ketika item chat diklik, akan muncul `Toast` yang menampilkan isi pesannya.

```
init {
    itemView.setOnClickListener {
        val position = bindingAdapterPosition
        if (position != RecyclerView.NO_POSITION) {
            val chat = chatList[position]
            val toastText = "${chat.sender} bilang: \"${chat.message}\""
            Toast.makeText(itemView.context, toastText,
Toast.LENGTH_SHORT).show()
        }
    }
}
```

Fungsi Adapter

→ Membuat tampilan untuk setiap item menggunakan `item_chat.xml`.

```
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int):
ChatViewHolder {
```

→ Mengatur isi teks untuk masing-masing item chat.

```
override fun onBindViewHolder(holder: ChatViewHolder, position: Int) {
```

→ Mengembalikan jumlah item yang akan ditampilkan.

```
override fun getItemCount(): Int = chatList.size
```

6. ChatMessage.kt

→ Data class ini mendefinisikan struktur dari setiap pesan chat:

- sender: siapa pengirimnya (User/Ahli Gizi)
- message: isi pesan
- time: waktu pengiriman pesan

```
package com.example.tigapuluhharisehat.model

data class ChatMessage(
    val sender: String,
    val message: String,
    val time: String
)
```