code-to-pdf

Index of Files

.pdfignore

Location: .

...

```
*.iml
.kotlin
.gradle
**/build/
xcuserdata
!src/**/build/
local.properties
.idea
.DS_Store
captures
.externalNativeBuild
.cxx
*.xcodeproj/*
!*.xcodeproj/project.pbxproj
!*.xcodeproj/xcshareddata/
!*.xcodeproj/project.xcworkspace/
!*.xcworkspace/contents.xcworkspacedata
**/xcshareddata/WorkspaceSettings.xcsettings
.venv
.idea
code-to-pdf.py
```

NetworkClient.kt

Location: example/

...

```kotlin
package org.example.core.network

import io.ktor.client.HttpClient
import io.ktor.client.plugins.auth.Auth
import io.ktor.client.plugins.auth.providers.BearerTokens
import io.ktor.client.plugins.auth.providers.bearer
import io.ktor.client.plugins.contentnegotiation.ContentNegotiation
import io.ktor.client.plugins.logging.LogLevel
import io.ktor.client.plugins.logging.Logging
import io.ktor.client.request.get
import io.ktor.client.request.post
import io.ktor.client.request.setBody
import io.ktor.client.statement.HttpResponse
import io.ktor.http.ContentType
import io.ktor.http.appendPathSegments
import io.ktor.http.contentType
import io.ktor.http.encodedPath
import io.ktor.serialization.kotlinx.json.json
import kotlinx.serialization.json.Json
import org.example.core.local.ValuesStoreManager


class NetworkClient(
    val baseUrl: String,
    private val valuesStoreManager: ValuesStoreManager
) {
    private val ktorClient by lazy {
        HttpClient {
            install(ContentNegotiation) {
                json(
                    Json {
                        ignoreUnknownKeys = true
                        isLenient = true
                    }
                )
            }

            install(Logging) {
                level = LogLevel.ALL
            }

            install(Auth) {
                bearer {
                    loadTokens {
                        BearerTokens(valuesStoreManager.token, "")
                    }

                    sendWithoutRequest { request ->
                        val allowedBearer = listOf(
                            "/qris",
                            "/api/v1/product",
                            "/api/v1/product/prepaid",
                            "/api/v1/categories"
                        )
                        allowedBearer.contains(request.url.encodedPath)
                    }
                }
            }
        }
    }

    suspend fun get(path: String, parameter: Map<String, String>? = null): HttpResponse {
        return ktorClient.get(baseUrl) {
            url {
                appendPathSegments(path)
                parameter?.forEach { param ->
                    parameters.append(param.key, param.value)
                }
            }
            contentType(ContentType.Application.Json)
```

```
        }
    }

    suspend fun post(
        path: String,
        parameter: Map<String, String>? = null,
        body: Any? = null
    ): HttpResponse {
        return ktorClient.post(baseUrl) {
            url {
                appendPathSegments(path)
                parameter?.forEach { param ->
                    parameters.append(param.key, param.value)
                }
            }
            contentType(ContentType.Application.Json)
            setBody(body)
        }
    }
}
```

NetworkRepository.kt

Location: example/

...

```kotlin
package org.example.core.network

import io.ktor.client.call.body
import io.ktor.client.statement.HttpResponse
import io.ktor.client.statement.bodyAsText
import io.ktor.http.isSuccess
import io.ktor.serialization.JsonConvertException
import io.ktor.serialization.kotlinx.json.DefaultJson
import kotlinx.coroutines.delay
import kotlinx.coroutines.flow.Flow
import kotlinx.coroutines.flow.catch
import kotlinx.coroutines.flow.flow
import kotlinx.coroutines.flow.onStart
import kotlinx.serialization.json.jsonObject
import kotlinx.serialization.json.jsonPrimitive

abstract class NetworkRepository {


    protected inline fun <reified T, U>(suspend () -> HttpResponse).reduce(
        crossinline block: (T) -> Async<U>
    ) : Flow<Async<U>> {
        return flow {
            val httpResponse = invoke()
            if (httpResponse.status.isSuccess()) {
                val data = httpResponse.body<T>()
                val dataState = block.invoke(data)
                emit(dataState)
            } else {
                val message = try {
                    val json = DefaultJson
                        .parseToJsonElement(httpResponse.bodyAsText())
                        .jsonObject
                    json["message"]?.jsonPrimitive?.content
                } catch (e: Throwable) {
                    e.printStackTrace()
                    httpResponse.bodyAsText()
                }
                val throwable = Throwable(message)
                val state = Async.Failure(throwable)
                emit(state)
            }
        }.onStart {
            emit(Async.Loading)
        }.catch {
            val state = Async.Failure(it)
            emit(state)
        }
    }
}
```

libs.versions.toml

Location: example/example_again/

...

```
[versions]
agp = "8.2.2"
android-compileSdk = "34"
android-minSdk = "25"
android-targetSdk = "34"
androidx-activityCompose = "1.9.1"
androidx-appcompat = "1.7.0"
androidx-constraintlayout = "2.1.4"
androidx-core-ktx = "1.13.1"
androidx-espresso-core = "3.6.1"
androidx-lifecycle = "2.8.0"
androidx-material = "1.12.0"
androidx-test-junit = "1.2.1"
circuitFoundation = "0.23.1"
circuitxGestureNavigation = "0.23.1"
compose-plugin = "1.6.11"
imageLoader = "1.8.2"
junit = "4.13.2"
kotlin = "2.0.20"
kotlinxCoroutinesCore = "1.9.0-RC.2"
ktorClientCore = "3.0.0-beta-2"
multiplatformSettingsNoArg = "1.1.1"
jetbrainsKotlinJvm = "1.9.0"

[libraries]
circuitx-gesture-navigation = { module = "com.slack.circuit:circuitx-gesture-navigation", version.ref = "circuitxGestureNavigation" }
circuit-foundation = { module = "com.slack.circuit:circuit-foundation", version.ref = "circuitFoundation" }
image-loader = { module = "io.github.qdsfdhvh:image-loader", version.ref = "imageLoader" }
kotlin-test = { module = "org.jetbrains.kotlin:kotlin-test", version.ref = "kotlin" }
kotlin-test-junit = { module = "org.jetbrains.kotlin:kotlin-test-junit", version.ref = "kotlin" }
junit = { group = "junit", name = "junit", version.ref = "junit" }
androidx-core-ktx = { group = "androidx.core", name = "core-ktx", version.ref = "androidx-core-ktx" }
androidx-test-junit = { group = "androidx.test.ext", name = "junit", version.ref = "androidx-test-junit" }
androidx-espresso-core = { group = "androidx.test.espresso", name = "espresso-core", version.ref = "androidx-espresso-core" }
androidx-appcompat = { group = "androidx.appcompat", name = "appcompat", version.ref = "androidx-appcompat" }
androidx-material = { group = "com.google.android.material", name = "material", version.ref = "androidx-material" }
androidx-constraintlayout = { group = "androidx.constraintlayout", name = "constraintlayout", version.ref = "androidx-constraintlayout" }
androidx-activity-compose = { module = "androidx.activity:activity-compose", version.ref = "androidx-activityCompose" }
androidx-lifecycle-viewmodel = { group = "org.jetbrains.androidx.lifecycle", name = "lifecycle-viewmodel", version.ref = "androidx-lifecycle" }
androidx-lifecycle-runtime-compose = { group = "org.jetbrains.androidx.lifecycle", name = "lifecycle-runtime-compose", version.ref = "androidx-lifecycle" }
kotlinx-coroutines-android = { module = "org.jetbrains.kotlinx:kotlinx-coroutines-android", version.ref = "kotlinxCoroutinesCore" }
kotlinx-coroutines-core = { module = "org.jetbrains.kotlinx:kotlinx-coroutines-core", version.ref = "kotlinxCoroutinesCore" }
ktor-client-content-negotiation = { module = "io.ktor:ktor-client-content-negotiation", version.ref = "ktorClientCore" }
ktor-client-core = { module = "io.ktor:ktor-client-core", version.ref = "ktorClientCore" }
```

```
ktor-client-darwin = { module = "io.ktor:ktor-client-darwin", version.ref = "ktorClientCore" }
ktor-client-logging = { module = "io.ktor:ktor-client-logging", version.ref = "ktorClientCore" }
ktor-client-okhttp = { module = "io.ktor:ktor-client-okhttp", version.ref = "ktorClientCore" }
ktor-client-auth = { module = "io.ktor:ktor-client-auth", version.ref = "ktorClientCore" }
ktor-serialization-kotlinx-json = { module = "io.ktor:ktor-serialization-kotlinx-json", version.ref = "ktorClientCore" }
multiplatform-settings = { module = "com.russhwolf:multiplatform-settings", version.ref = "multiplatformSettingsNoArg" }
multiplatform-settings-no-arg = { module = "com.russhwolf:multiplatform-settings-no-arg", version.ref = "multiplatformSettingsNoArg" }

[plugins]
androidApplication = { id = "com.android.application", version.ref = "agp" }
androidLibrary = { id = "com.android.library", version.ref = "agp" }
jetbrainsCompose = { id = "org.jetbrains.compose", version.ref = "compose-plugin" }
compose-compiler = { id = "org.jetbrains.kotlin.plugin.compose", version.ref = "kotlin" }
kotlinMultiplatform = { id = "org.jetbrains.kotlin.multiplatform", version.ref = "kotlin" }
kotlinSerializer = { id = "org.jetbrains.kotlin.plugin.serialization", version.ref = "kotlin" }
nativeCocoaPod = { id = "org.jetbrains.kotlin.native.cocoapods", version.ref = "kotlin" }
kotlinAndroidParcelize = { id = "org.jetbrains.kotlin.plugin.parcelize", version.ref = "kotlin" }
jetbrains-kotlin-jvm = { id = "org.jetbrains.kotlin.jvm", version.ref = "jetbrainsKotlinJvm" }
```

requirements.txt

Location: .

...

```
  1: arabic-reshaper==3.0.0
  2: asn1crypto==1.5.1
  3: certifi==2024.8.30
  4: cffi==1.17.0
  5: chardet==5.2.0
  6: charset-normalizer==3.3.2
  7: click==8.1.7
  8: cryptography==43.0.0
  9: cssselect2==0.7.0
 10: defusedxml==0.7.1
 11: fonttools==4.53.1
 12: fpdf2==2.7.9
 13: html5lib==1.1
 14: idna==3.8
 15: lxml==5.3.0
 16: oscrypto==1.3.0
 17: pillow==10.4.0
 18: pycparser==2.22
 19: Pygments==2.18.0
 20: pyHanko==0.25.1
 21: pyhanko-certvalidator==0.26.3
 22: pypdf==4.3.1
 23: PyPDF2==3.0.1
 24: pypng==0.20220715.0
 25: python-bidi==0.6.0
 26: PyYAML==6.0.2
 27: qrcode==7.4.2
 28: reportlab==4.2.2
 29: requests==2.32.3
 30: six==1.16.0
 31: svglib==1.5.1
 32: tinycss2==1.3.0
 33: typing_extensions==4.12.2
 34: tzlocal==5.2
 35: uritools==4.0.3
 36: urllib3==2.2.2
 37: webencodings==0.5.1
 38: xhtml2pdf==0.2.16
```