

Tổng hợp mã nguồn dự án: AnTamViecLam

.gitignore

<<<<< HEAD

local.properties

>>>>> origin/main

README.md

AnTamViecLam

<<<<< HEAD

=====

``

```
├── data/
│   ├── model/          // Data classes (User.kt, Job.kt, Review.kt...)
│   ├── network/        // Các lớp tương tác với Firebase
│   │   ├── FirestoreClient.kt
│   │   └── FirebaseAuthClient.kt
│   └── repository/    // Nơi tổng hợp dữ liệu từ các nguồn
│       ├── AuthRepository.kt
│       ├── JobRepository.kt
│       ├── UserRepository.kt
│       └── impl/          // Các lớp implementation của interface trên
│           ├── AuthRepositoryImpl.kt
│           ├── JobRepositoryImpl.kt
│           └── UserRepositoryImpl.kt
|
├── di/              // Dependency Injection (Hilt or Koin)
│   └── AppModule.kt  // Cung cấp các dependency chung
|
└── ui/             // Chứa toàn bộ UI, chia theo từng feature
    ├── auth/          // Feature Đăng ký, Đăng nhập
    │   ├── LoginFragment.kt
    │   ├── RegisterFragment.kt
    │   └── AuthViewModel.kt
    |
    └── main/          // Màn hình chính chứa Bottom Navigation
        └── MainActivity.kt
```

```
|  
|   └── home/          // Feature Trang chủ (danh sách công việc)  
|       ├── HomeFragment.kt  
|       ├── HomeViewModel.kt  
|       └── adapter/  
|             └── JobAdapter.kt  
  
|   └── map/           // Feature Bản đồ công việc  
|       ├── MapViewFragment.kt  
|       └── MapViewModel.kt  
  
|   └── job_details/    // Feature Chi tiết công việc  
|       ├── JobDetailsFragment.kt  
|       └── JobDetailsViewModel.kt  
  
|   └── profile/        // Feature Quản lý hồ sơ  
|       ├── ProfileFragment.kt  
|       ├── EditProfileFragment.kt  
|       └── ProfileViewModel.kt  
  
|   └── bhhx/           // Feature BHXH  
|       ├── BhhxFragment.kt  
|       ├── BhhxViewModel.kt  
|       └── BhhxInfoDetailsFragment.kt  
  
|   └── chat/           // Feature Chat  
|       ├── ChatListFragment.kt // Danh sách các cuộc trò chuyện  
|       ├── ChatDetailFragment.kt // Màn hình chat 1-1  
|       └── ChatViewModel.kt  
  
|   └── posting/         // Feature Đăng tin của NTD  
|       ├── CreateJobFragment.kt  
|       └── CreateJobViewModel.kt  
  
|   └── base/            // Các lớp base cho Activity, Fragment, ViewModel  
|       ├── BaseActivity.kt  
|       └── BaseViewModel.kt  
  
└── utils/             // Các lớp tiện ích dùng chung  
    ├── Constants.kt      // Chứa các hằng số  
    ├── Extensions.kt     // Các extension function (e.g., View.show())  
    ├── LocationHelper.kt  // Lớp helper xử lý location  
    └── DateTimeUtils.kt   // Lớp helper xử lý ngày giờ
```

```

```
>>>>> 78d0e2f10b7c9453339a3b4ce51be88ed643f0a5
```

## build.gradle.kts

```
// Top-level build file where you can add configuration options common to all sub-
projects/modules.
plugins {
 alias(libs.plugins.android.application) apply false
 alias(libs.plugins.kotlin.android) apply false
 // alias(libs.plugins.kotlin.compose) apply false
 alias(libs.plugins.android.library) apply false
 alias(libs.plugins.hilt) apply false
 alias(libs.plugins.ksp) apply false
 alias(libs.plugins.google.services) apply false
 alias(libs.plugins.firebaseio.crashlytics) apply false
 alias(libs.plugins.android.secrets.gradle.plugin) apply false

}
```

## gradle.properties

```
Project-wide Gradle settings.
IDE (e.g. Android Studio) users:
Gradle settings configured through the IDE *will override*
any settings specified in this file.
For more details on how to configure your build environment visit
http://www.gradle.org/docs/current/userguide/build_environment.html
Specifies the JVM arguments used for the daemon process.
The setting is particularly useful for tweaking memory settings.
org.gradle.jvmargs=-Xmx2048m -Dfile.encoding=UTF-8
When configured, Gradle will run in incubating parallel mode.
This option should only be used with decoupled projects. For more details, visit
https://developer.android.com/r/tools/gradle-multi-project-decoupled-projects
org.gradle.parallel=true
AndroidX package structure to make it clearer which packages are bundled with the
Android operating system, and which are packaged with your app's APK
https://developer.android.com/topic/libraries/support-library/androidx-rn
android.useAndroidX=true
Kotlin code style for this project: "official" or "obsolete":
kotlin.code.style=official
Enables namespacing of each library's R class so that its R class includes only the
resources declared in the library itself and none from the library's dependencies,
thereby reducing the size of the R class for that library
```

```
android.nonTransitiveRClass=true
android.useAndroidX=true
android.enableJetifier=true
org.gradle.jvmargs=-Xmx4096m
org.gradle.parallel=true
org.gradle.caching=true

gradlew
#!/bin/sh

#
Copyright © 2015 the original authors.
#
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
#
https://www.apache.org/licenses/LICENSE-2.0
#
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
#
SPDX-License-Identifier: Apache-2.0
#

#####
#####
#
Gradle start up script for POSIX generated by Gradle.
#
Important for running:
#
(1) You need a POSIX-compliant shell to run this script. If your /bin/sh is
noncompliant, but you have some other compliant shell such as ksh or
bash, then to run this script, type that shell name before the whole
command line, like:
#
ksh Gradle
#
Busybox and similar reduced shells will NOT work, because this script
```

```

requires all of these POSIX shell features:
* functions;
* expansions «$var», «${var}», «${var:-default}», «${var+SET}»,
«${var#prefix}», «${var%suffix}», and «$(cmd)»;
* compound commands having a testable exit status, especially «case»;
* various built-in commands including «command», «set», and «ulimit».
#
Important for patching:
#
(2) This script targets any POSIX shell, so it avoids extensions provided
by Bash, Ksh, etc; in particular arrays are avoided.
#
The "traditional" practice of packing multiple parameters into a
space-separated string is a well documented source of bugs and security
problems, so this is (mostly) avoided, by progressively accumulating
options in "$@", and eventually passing that to Java.
#
Where the inherited environment variables (DEFAULT_JVM_OPTS, JAVA_OPTS,
and GRADLE_OPTS) rely on word-splitting, this is performed explicitly;
see the in-line comments for details.
#
There are tweaks for specific operating systems such as AIX, CygWin,
Darwin, MinGW, and NonStop.
#
(3) This script is generated from the Groovy template
https://github.com/gradle/gradle/blob/HEAD/platforms/jvm/plugins-
application/src/main/resources/org/gradle/api/internal/plugins/unixStartScript.txt
within the Gradle project.
#
You can find Gradle at https://github.com/gradle/.
#
#####
#####

Attempt to set APP_HOME

Resolve links: $0 may be a link
app_path=$0

Need this for daisy-chained symlinks.
while
 APP_HOME=${app_path%"${app_path##*/}"} # leaves a trailing /; empty if no leading
path

```

```

[-h "$app_path"]
do
ls=$(ls -ld "$app_path")
link=${ls##*' -> '}
case $link in #(
/*) app_path=$link ;;
*) app_path=APP_HOMElink ;;
esac
done

This is normally unused
shellcheck disable=SC2034
APP_BASE_NAME=${0##*/}

Discard cd standard output in case $CDPATH is set
#(https://github.com/gradle/gradle/issues/25036)
APP_HOME=$(cd -P "${APP_HOME:-.}" > /dev/null && printf '%s\n' "$PWD") || exit

Use the maximum available, or set MAX_FD != -1 to use that value.
MAX_FD=maximum

warn () {
 echo "$*"
} >&2

die () {
 echo
 echo "$*"
 echo
 exit 1
} >&2

OS specific support (must be 'true' or 'false').
cygwin=false
msys=false
darwin=false
nonstop=false
case "$(uname)" in #(
 CYGWIN*) cygwin=true ;;
 Darwin*) darwin=true ;;
 MSYS* | MINGW*) msys=true ;;
 NONSTOP*) nonstop=true ;;
esac

```

```

CLASSPATH="\\\"\\\""

Determine the Java command to use to start the JVM.
if [-n "$JAVA_HOME"] ; then
 if [-x "$JAVA_HOME/jre/sh/java"] ; then
 # IBM's JDK on AIX uses strange locations for the executables
 JAVACMD=$JAVA_HOME/jre/sh/java
 else
 JAVACMD=$JAVA_HOME/bin/java
 fi
 if [! -x "$JAVACMD"] ; then
 die "ERROR: JAVA_HOME is set to an invalid directory: $JAVA_HOME
Please set the JAVA_HOME variable in your environment to match the
location of your Java installation."
 fi
else
 JAVACMD=java
 if ! command -v java >/dev/null 2>&1
 then
 die "ERROR: JAVA_HOME is not set and no 'java' command could be found in your
PATH.

Please set the JAVA_HOME variable in your environment to match the
location of your Java installation."
 fi
Increase the maximum file descriptors if we can.
if ! "$cygwin" && ! "$darwin" && ! "$nonstop" ; then
 case $MAX_FD in
 max*)
 # In POSIX sh, ulimit -H is undefined. That's why the result is checked to see if it
 worked.
 # shellcheck disable=SC2039,SC3045
 MAX_FD=$(ulimit -H -n) ||
 warn "Could not query maximum file descriptor limit"
 esac
 case $MAX_FD in
 " | soft) ;;; #(
 *)
 # In POSIX sh, ulimit -n is undefined. That's why the result is checked to see if it worked.
 esac
fi

```

```

shellcheck disable=SC2039,SC3045
ulimit -n "$MAX_FD" ||
warn "Could not set maximum file descriptor limit to $MAX_FD"
esac
fi

Collect all arguments for the java command, stacking in reverse order:
* args from the command line
* the main class name
* -classpath
* -D...appname settings
* --module-path (only if needed)
* DEFAULT_JVM_OPTS, JAVA_OPTS, and GRADLE_OPTS environment variables.

For Cygwin or MSYS, switch paths to Windows format before running java
if "$cygwin" || "$msys" ; then
 APP_HOME=$(cygpath --path --mixed "$APP_HOME")
 CLASSPATH=$(cygpath --path --mixed "$CLASSPATH")

 JAVACMD=$(cygpath --unix "$JAVACMD")

Now convert the arguments - kludge to limit ourselves to /bin/sh
for arg do
 if
 case $arg in
 -*)
 false ;;
 /*?*)
 t=${arg#/} t=${t%/*}
 # looks like a POSIX filepath
 [-e "$t"] ;;
 *)
 false ;;
 esac
 then
 arg=$(cygpath --path --ignore --mixed "$arg")
 fi
 # Roll the args list around exactly as many times as the number of
 # args, so each arg winds up back in the position where it started, but
 # possibly modified.
 #
 # NB: a `for` loop captures its iteration list before it begins, so
 # changing the positional parameters here affects neither the number of
 # iterations, nor the values presented in `arg`.
 shift # remove old arg
 set -- "$@" "$arg" # push replacement arg
done

```

```

fi

Add default JVM options here. You can also use JAVA_OPTS and GRADLE_OPTS to pass
JVM options to this script.
DEFAULT_JVM_OPTS="-Xmx64m" "-Xms64m"

Collect all arguments for the java command:
* DEFAULT_JVM_OPTS, JAVA_OPTS, and optsEnvironmentVar are not allowed to contain
shell fragments,
and any embedded shellness will be escaped.
* For example: A user cannot expect ${Hostname} to be expanded, as it is an
environment variable and will be
treated as '${Hostname}' itself on the command line.

set -- \
 "-Dorg.gradle.appname=$APP_BASE_NAME" \
 -classpath "$CLASSPATH" \
 -jar "$APP_HOME/gradle/wrapper/gradle-wrapper.jar" \
 "$@"

Stop when "xargs" is not available.
if ! command -v xargs >/dev/null 2>&1
then
 die "xargs is not available"
fi

Use "xargs" to parse quoted args.
#
With -n1 it outputs one arg per line, with the quotes and backslashes removed.
#
In Bash we could simply go:
#
readarray ARGS < <(xargs -n1 <<< "$var") &&
set -- "${ARGS[@]}" "$@"
#
but POSIX shell has neither arrays nor command substitution, so instead we
post-process each arg (as a line of input to sed) to backslash-escape any
character that might be a shell metacharacter, then use eval to reverse
that process (while maintaining the separation between arguments), and wrap
the whole thing up as a single "set" statement.
#
This will of course break if any of these variables contains a newline or

```

```

an unmatched quote.
#
eval "set -- $(
 printf '%s\n' "$DEFAULT_JVM_OPTS $JAVA_OPTS $GRADLE_OPTS" |
 xargs -n1 |
 sed 's~[^-[:alnum:]+,./:=@_]~\\&~g;' |
 tr '\n' ''
)" "$@"
exec "$JAVACMD" "$@"

```

### **gradlew.bat**

```

@rem
@rem Copyright 2015 the original author or authors.
@rem
@rem Licensed under the Apache License, Version 2.0 (the "License");
@rem you may not use this file except in compliance with the License.
@rem You may obtain a copy of the License at
@rem
@rem https://www.apache.org/licenses/LICENSE-2.0
@rem
@rem Unless required by applicable law or agreed to in writing, software
@rem distributed under the License is distributed on an "AS IS" BASIS,
@rem WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
@rem See the License for the specific language governing permissions and
@rem limitations under the License.
@rem
@rem SPDX-License-Identifier: Apache-2.0
@rem

@if "%DEBUG%"=="" @echo off
@rem
#####
@rem Gradle startup script for Windows
@rem
#####
#####
```

```
@rem Set local scope for the variables with windows NT shell
if "%OS%"=="Windows_NT" setlocal

set DIRNAME=%~dp0
if "%DIRNAME%"=="" set DIRNAME=.
@rem This is normally unused
set APP_BASE_NAME=%~n0
set APP_HOME=%DIRNAME%

@rem Resolve any "." and ".." in APP_HOME to make it shorter.
for %%i in ("%APP_HOME%") do set APP_HOME=%%~fi

@rem Add default JVM options here. You can also use JAVA_OPTS and GRADLE_OPTS to
pass JVM options to this script.
set DEFAULT_JVM_OPTS="-Xmx64m" "-Xms64m"

@rem Find java.exe
if defined JAVA_HOME goto findJavaFromJavaHome

set JAVA_EXE=java.exe
%JAVA_EXE% -version >NUL 2>&1
if %ERRORLEVEL% equ 0 goto execute

echo. 1>&2
echo ERROR: JAVA_HOME is not set and no 'java' command could be found in your PATH.
1>&2
echo. 1>&2
echo Please set the JAVA_HOME variable in your environment to match the 1>&2
echo location of your Java installation. 1>&2

goto fail

:findJavaFromJavaHome
set JAVA_HOME=%JAVA_HOME:"=%
set JAVA_EXE=%JAVA_HOME%/bin/java.exe

if exist "%JAVA_EXE%" goto execute

echo. 1>&2
echo ERROR: JAVA_HOME is set to an invalid directory: %JAVA_HOME% 1>&2
echo. 1>&2
echo Please set the JAVA_HOME variable in your environment to match the 1>&2
echo location of your Java installation. 1>&2
```

```

goto fail

:execute
@rem Setup the command line

set CLASSPATH=

@rem Execute Gradle
"%JAVA_EXE%" %DEFAULT_JVM_OPTS% %JAVA_OPTS% %GRADLE_OPTS% -
Dorg.gradle.appname=%APP_BASE_NAME%" -classpath "%CLASSPATH%" -jar
"%APP_HOME%\gradle\wrapper\gradle-wrapper.jar" %*

:end
@rem End local scope for the variables with windows NT shell
if %ERRORLEVEL% equ 0 goto mainEnd

:fail
rem Set variable GRADLE_EXIT_CONSOLE if you need the _script_ return code instead of
rem the _cmd.exe /c_ return code!
set EXIT_CODE=%ERRORLEVEL%
if %EXIT_CODE% equ 0 set EXIT_CODE=1
if not ""=="%GRADLE_EXIT_CONSOLE%" exit %EXIT_CODE%
exit /b %EXIT_CODE%

:mainEnd
if "%OS%"=="Windows_NT" endlocal

:omega

```

## **local.properties**

```

This file is automatically generated by Android Studio.
Do not modify this file -- YOUR CHANGES WILL BE ERASED!
#
This file should *NOT* be checked into Version Control Systems,
as it contains information specific to your local configuration.
#
Location of the SDK. This is only used by Gradle.
For customization when using a Version Control System, please read the
header note.
sdk.dir=C:\\\\Users\\\\ADMIN\\\\AppData\\\\Local\\\\Android\\\\Sdk

```

```
CLOUDINARY_CLOUD_NAME=diutnceax
CLOUDINARY_API_KEY=852589725185877
CLOUDINARY_API_SECRET=dyYSiYgvUsn3Tx6HWe47RD1WJNA
```

### settings.gradle.kts

```
pluginManagement {
 repositories {
 google()
 mavenCentral()
 gradlePluginPortal()
 }
}

dependencyResolutionManagement {
 repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
 repositories {
 google()
 mavenCentral()
 maven {
 url = uri("https://maven.track-asia.com/repository/maven-public/")
 }
 // If you need to add a custom Maven repository, you would add it here like this:
 // maven {
 // url = uri("https://your.custom.repository/maven2/")
 // }
 }
}
rootProject.name = "AnTamViecLam"
include(":app")
```

### app/.gitignore

```
/build
```

### app/build.gradle.kts

```
// Đảm bảo 2 dòng import này ở đầu file
import java.util.Properties
import java.io.FileInputStream
// ĐOẠN CODE LOGIC MỚI
val localProperties = Properties()
val localPropertiesFile = rootProject.file("local.properties")
if (localPropertiesFile.exists()) {
 localProperties.load(FileInputStream(localPropertiesFile))
}
```

```
// Đọc 3 giá trị mới
val cloudName = localProperties.getProperty("CLOUDINARY_CLOUD_NAME") ?: ""
val apiKey = localProperties.getProperty("CLOUDINARY_API_KEY") ?: ""
val apiSecret = localProperties.getProperty("CLOUDINARY_API_SECRET") ?: ""

plugins {
 alias(libs.plugins.android.application)
 alias(libs.plugins.kotlin.android)
 alias(libs.plugins.ksp)
 alias(libs.plugins.firebaseio.crashlytics)
 alias(libs.plugins.kotlin.compose.compiler)
 alias(libs.plugins.android.secrets.gradle.plugin)
 id("com.google.dagger.hilt.android")
 id("com.google.gms.google-services")
}

android {
 namespace = "com.example.antamvieclam"
 compileSdk = 34

 defaultConfig {
 applicationId = "com.example.antamvieclam"
 minSdk = 24
 targetSdk = 34
 versionCode = 1
 versionName = "1.0"
 testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
 }

 // Chỉ giữ lại dòng buildConfigField ở đây
 buildConfigField("String", "CLOUDINARY_CLOUD_NAME", "\"$cloudName\"")
 buildConfigField("String", "CLOUDINARY_API_KEY", "\"$apiKey\"")
 buildConfigField("String", "CLOUDINARY_API_SECRET", "\"$apiSecret\"") }

buildTypes {
 release {
 isMinifyEnabled = false
 proguardFiles(
 getDefaultProguardFile("proguard-android-optimize.txt"),
 "proguard-rules.pro"
)
 }
}
```

```
compileOptions {
 sourceCompatibility = JavaVersion.VERSION_1_8
 targetCompatibility = JavaVersion.VERSION_1_8
}

kotlinOptions {
 jvmTarget = "1.8"
}
// kotlin {
// jvmToolchain(17)
// }
buildFeatures {
 compose = true
 buildConfig = true
}
}

dependencies {
 implementation(libs.androidx.navigation.compose)
 implementation("androidx.compose.material:material-icons-extended:1.7.5")
 implementation(libs.androidx.hilt.navigation.compose)
 implementation(libs.androidx.lifecycle.runtime.compose)
 implementation(libs.coil.compose)

 // Jetpack Compose BoM (Bill of Materials) - Giúp quản lý phiên bản các thư viện Compose
 val composeBom = platform("androidx.compose:compose-bom:2024.05.00") // Kiểm tra phiên bản mới nhất
 implementation(composeBom)
 androidTestImplementation(composeBom)

 // Các thư viện Compose cần thiết
 implementation("androidx.compose.ui:ui")
 implementation("androidx.compose.ui:ui-graphics")
 implementation("androidx.compose.ui:ui-tooling-preview")
 implementation("androidx.compose.material3:material3")

 // Thư viện cần thiết cho setContent
 implementation("androidx.activity:activity-compose:1.9.0") // Kiểm tra phiên bản mới nhất
```

```
// TrackAsia Core SDK
 implementation("io.github.track-asia:android-sdk:2.0.1")
// TrackAsia Data Models
 implementation("io.github.track-asia:android-sdk-geojson:2.0.1")
 implementation("io.github.track-asia:android-sdk-turf:2.0.1")
// TrackAsia Plugins
 implementation("io.github.track-asia:android-plugin-annotation-v9:2.0.1")
// TrackAsia Navigation
 implementation("io.github.track-asia:libandroid-navigation:2.0.0")
 implementation("io.github.track-asia:libandroid-navigation-ui:2.0.0")
// Location Services
 implementation("com.google.android.gms:play-services-location:21.0.1")

// Phần này của bạn đã đúng, giữ nguyên
 implementation(libs.cloudinary.android)
 implementation(libs.androidx.core.ktx)
 implementation(platform(libs.androidx.compose.bom))
 implementation(libs.androidx.compose.ui)
 implementation(libs.androidx.compose.ui.graphics)
 implementation(libs.androidx.compose.ui.tooling.preview)
 implementation(libs.androidx.compose.material3)
 implementation(libs.hilt.android)
 ksp(libs.hilt.compiler)
 implementation(libs.androidx.room.runtime)
 implementation(libs.androidx.room.ktx)
 ksp(libs.androidx.room.compiler)
 implementation(platform(libs.firebaseio.bom))
 implementation(libs.firebaseio.auth.ktx)
 implementation(libs.firebaseio.firestore.ktx)
 implementation(libs.firebaseio.storage.ktx)
 implementation(libs.firebaseio.messaging.ktx)
 implementation(libs.firebaseio.crashlytics.ktx)
 implementation(libs.play.services.auth)
 implementation(libs.kotlinx.coroutines.core)
 implementation(libs.kotlinx.coroutines.android)
 implementation(libs.retrofit)
 implementation(libs.converter.gson)
 implementation(libs.logging.interceptor)
 testImplementation(libs.junit)
 androidTestImplementation(libs.androidx.junit)
 androidTestImplementation(libs.androidx.espresso.core)
 androidTestImplementation(platform(libs.androidx.compose.bom))
 debugImplementation(libs.androidx.compose.ui.tooling)
```

```
}
```

```
ksp {
 arg("hilt.CorrectErrorTypes", "true")
}
```

### app/google-services.json

```
{
 "project_info": {
 "project_number": "221272132411",
 "project_id": "antamvieclam",
 "storage_bucket": "antamvieclam.firebaseiostorage.app"
 },
 "client": [
 {
 "client_info": {
 "mobilesdk_app_id": "1:221272132411:android:c432fab2ad267434c8451e",
 "android_client_info": {
 "package_name": "com.example.antamvieclam"
 }
 },
 "oauth_client": [
 {
 "client_id": "221272132411-826fghoh93hedhe8598r13fankndh3m.apps.googleusercontent.com",
 "client_type": 1,
 "android_info": {
 "package_name": "com.example.antamvieclam",
 "certificate_hash": "d874dcf8d6a76c39867094613e7648c3f4854e1e"
 }
 },
 {
 "client_id": "221272132411-q7mkd1kiqvln71k4rjvlm2567vloqci9.apps.googleusercontent.com",
 "client_type": 3
 }
],
 "api_key": [
 {
 "current_key": "AIzaSyCuyB9Bi3hWhR3JCuaZ4kZPq1NZ8u6-vEs"
 }
],
 "services": {

```

```
"appinvite_service": {
 "other_platform_oauth_client": [
 {
 "client_id": "221272132411-q7mkd1kiqvln71k4rjvlm2567vloqci9.apps.googleusercontent.com",
 "client_type": 3
 }
]
},
"configuration_version": "1"
}
```

### [app/proguard-rules.pro](#)

```
Add project specific ProGuard rules here.
You can control the set of applied configuration files using the
proguardFiles setting in build.gradle.
#
For more details, see
http://developer.android.com/guide/developing/tools/proguard.html

If your project uses WebView with JS, uncomment the following
and specify the fully qualified class name to the JavaScript interface
class:
#-keepclassmembers class fqcn.of.javascript.interface.for.webview {
public *;
#}

Uncomment this to preserve the line number information for
debugging stack traces.
#-keepattributes SourceFile,LineNumberTable

If you keep the line number information, uncomment this to
hide the original source file name.
#-renamesourcefileattribute SourceFile
```

### [app/src/androidTest/java/com/example/antamvieclam/ExampleInstrumentedTest.kt](#)

```
package com.example.antamvieclam

import androidx.test.platform.app.InstrumentationRegistry
```

```
import androidx.test.ext.junit.runners.AndroidJUnit4

import org.junit.Test
import org.junit.runner.RunWith

import org.junit.Assert.*

/**
 * Instrumented test, which will execute on an Android device.
 *
 * See [testing documentation](http://d.android.com/tools/testing).
 */
@RunWith(AndroidJUnit4::class)
class ExampleInstrumentedTest {
 @Test
 fun useApplicationContext() {
 // Context of the app under test.
 val applicationContext = InstrumentationRegistry.getInstrumentation().targetContext
 assertEquals("com.example.antamvieclam", applicationContext.packageName)
 }
}
```

### [app/src/main/AndroidManifest.xml](#)

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:tools="http://schemas.android.com/tools">

 <application
 android:allowBackup="true"
 android:dataExtractionRules="@xml/data_extraction_rules"
 android:fullBackupContent="@xml/backup_rules"
 android:icon="@mipmap/ic_launcher"
 android:name="com.example.antamvieclam.AnVuiViecApplication"
 android:label="@string/app_name"
 android:roundIcon="@mipmap/ic_launcher_round"
 android:supportsRtl="true"
 android:theme="@style/Theme.AnTamViecLam">
 <activity
 android:name="com.example.antamvieclam.MainActivity"
 android:exported="true"
 android:label="@string/app_name"
 android:theme="@style/Theme.AnTamViecLam">
 <intent-filter>
```

```
<action android:name="android.intent.action.MAIN" />

 <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>

</application>

</manifest>
```

### [app/src/main/java/com/example/antamvieclam/AnVuiViecApplication.kt](#)

```
package com.example.antamvieclam
```

```
import android.app.Application
import com.cloudinary.android.MediaManager
import com.example.antamvieclam.BuildConfig
import com.trackasia.android.TrackAsia
import dagger.hilt.android.HiltAndroidApp
```

```
@HiltAndroidApp
class AnVuiViecApplication : Application() {
```

```
 override fun onCreate() {
 super.onCreate()

 // --- Khởi tạo TrackAsia với style URL trực tiếp ---
 val styleUrl = "https://maps.track-
asia.com/styles/v1/streets.json?key=52fdb6b306931761836057e5580a05be7"
 TrackAsia.getInstance(applicationContext).equals(styleUrl)
```

```
 // --- Khởi tạo Cloudinary ---
 val config = mutableMapOf<String, String>()
 config["cloud_name"] = BuildConfig.CLOUDINARY_CLOUD_NAME
 config["api_key"] = BuildConfig.CLOUDINARY_API_KEY
 config["api_secret"] = BuildConfig.CLOUDINARY_API_SECRET
 MediaManager.init(this, config)
}
```

### [app/src/main/java/com/example/antamvieclam/MainActivity.kt](#)

```
package com.example.antamvieclam
```

```
import android.os.Bundle
```

```
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.activity.enableEdgeToEdge
import com.example.antamvieclam.ui.navigation.AppNavigation // Import quan trọng
import com.example.antamvieclam.ui.theme.AnTamViecLamTheme
import dagger.hilt.android.AndroidEntryPoint // Import quan trọng

@AndroidEntryPoint // BẮT BUỘC: Đánh dấu Activity là một entry point cho Hilt
class MainActivity : ComponentActivity() {
 override fun onCreate(savedInstanceState: Bundle?) {
 super.onCreate(savedInstanceState)
 enableEdgeToEdge()
 setContent {
 AnTamViecLamTheme {
 // GỌI AppNavigation() để bắt đầu luồng của ứng dụng
 AppNavigation()
 }
 }
 }
}
```

#### [app/src/main/java/com/example/antamvieclam/data/model/Job.kt](#)

```
// app/src/main/java/com/example/antamvieclam/data/model/Job.kt
package com.example.antamvieclam.data.model
```

```
import com.google.firebase.firestore.GeoPoint
import com.google.firebase.firestore.ServerTimestamp
import java.util.Date
```

```
enum class PayType {
 PER_HOUR, PER_DAY, PER_PACKAGE
}
```

```
enum class JobStatus {
 OPEN, IN_PROGRESS, COMPLETED, CANCELLED
}
```

```
data class Job(
 var id: String = "",
 val employerId: String = "",
 val employerName: String = "",
 val employerProfileUrl: String? = null,
```

```

 val title: String = "",
 val description: String = "",
 val payRate: Double = 0.0,
 val payType: PayType = PayType.PER_HOUR,
 val location: GeoPoint? = null, // Vị trí chính xác trên bản đồ
 val addressString: String = "", // Địa chỉ dạng text để hiển thị

 var status: JobStatus = JobStatus.OPEN,
 var hiredWorkerId: String? = null,

 @ServerTimestamp
 val createdAt: Date? = null
)

// Dữ liệu cho đơn ứng tuyển
data class JobApplication(
 var id: String = "",
 val jobId: String = "",
 val applicantId: String = "", // workerId
 val employerId: String = "",
 val status: String = "PENDING", // PENDING, ACCEPTED, REJECTED
 @ServerTimestamp
 val appliedAt: Date? = null
)

```

[app/src/main/java/com/example/antamvieclam/data/model/User.kt](#)

```
package com.example.antamvieclam.data.model
```

```
import com.google.firebase.firestore.ServerTimestamp
import java.util.Date
```

```
// Phân loại người dùng
enum class UserType {
 WORKER, EMPLOYER
}
```

```
data class User(
 val uid: String = "",
 val phoneNumber: String? = "",
 val userType: UserType = UserType.WORKER,
 val fullName: String = "",
 var profileImageUrl: String? = null, // URL ảnh từ Cloudinary
 val address: String? = null,
```

```
// Thêm các trường khác theo cấu trúc của bạn
@ServerTimestamp
val createdAt: Date? = null
)
```

### **app/src/main/java/com/example/antamvieclam/data/repository/AuthRepository.kt**

```
package com.example.antamvieclam.data.repository
```

```
import com.google.firebase.auth.AuthCredential
```

```
// Đơn giản hóa sealed class
sealed class AuthResult {
 object Success : AuthResult()
 data class Error(val exception: Exception) : AuthResult()
}
```

```
interface AuthRepository {
 // Hàm mới để đăng nhập với Google credential
 suspend fun signInWithGoogle(credential: AuthCredential): AuthResult
 fun getCurrentUserId(): String?
 fun signOut()
}
```

### **app/src/main/java/com/example/antamvieclam/data/repository/JobRepository.kt**

```
// app/src/main/java/com/example/antamvieclam/data/repository/JobRepository.kt
package com.example.antamvieclam.data.repository
```

```
import com.example.antamvieclam.data.model.Job
import com.google.firebaseio.DocumentSnapshot
```

```
interface JobRepository {
 // Task 2.1
 suspend fun postJob(job: Job): Result<Unit>

 // Task 2.2
 suspend fun getJobs(lastVisibleJob: DocumentSnapshot?): Result<Pair<List<Job>, DocumentSnapshot?>>
 suspend fun getJobDetails(jobId: String): Result<Job>
 suspend fun applyForJob(jobId: String, employerId: String, applicantId: String): Result<Unit>
}
```

[\*\*app/src/main/java/com/example/antamvieclam/data/repository/UserRepository.kt\*\*](#)

```
package com.example.antamvieclam.data.repository
```

```
import android.net.Uri
import com.example.antamvieclam.data.model.User
```

```
sealed class ProfileResult {
 object Success : ProfileResult()
 data class Error(val message: String) : ProfileResult()
}
```

```
interface UserRepository {
 suspend fun createUserProfile(user: User, imageUri: Uri?): ProfileResult
 suspend fun getUserProfile(uid: String): User?
 // Thêm các hàm update, delete nếu cần
 suspend fun doesProfileExist(uid: String): Boolean
}
```

[\*\*app/src/main/java/com/example/antamvieclam/data/repository/impl/AuthRepositoryImpl.kt\*\*](#)

```
package com.example.antamvieclam.data.repository.impl
```

```
import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.AuthResult
import com.google.firebase.auth.AuthCredential
import com.google.firebase.auth.FirebaseAuth
import javax.inject.Inject
import kotlin.coroutines.resume
import kotlin.coroutines.suspendCoroutine
```

```
class AuthRepositoryImpl @Inject constructor(
 private val firebaseAuth: FirebaseAuth
) : AuthRepository {
```

```
 override suspend fun signInWithGoogle(credential: AuthCredential): AuthResult {
 return suspendCoroutine { continuation ->
 firebaseAuth.signInWithCredential(credential)
 .addOnCompleteListener { task ->
 if (task.isSuccessful) {
 continuation.resume(AuthResult.Success)
 } else {
```

```

 continuation.resume(AuthResult.Error(task.exception!!))
 }
}
}

override fun getCurrentUserId(): String? {
 return firebaseAuth.currentUser?.uid
}

override fun signOut() {
 firebaseAuth.signOut()
}
}

app/src/main/java/com/example/antamvieclam/data/repository/impl/JobRepositoryImpl.kt
//

app/src/main/java/com/example/antamvieclam/data/repository/impl/JobRepositoryImpl.kt
package com.example.antamvieclam.data.repository.impl

import com.example.antamvieclam.data.model.Job
import com.example.antamvieclam.data.model.JobApplication
import com.example.antamvieclam.data.repository.JobRepository
import com.google.firebaseio.firebaseio.DocumentSnapshot
import com.google.firebaseio.firebaseio.FirebaseFirestore
import com.google.firebaseio.firebaseio.Query
import kotlinx.coroutines.tasks.await
import javax.inject.Inject

class JobRepositoryImpl @Inject constructor(
 private val firestore: FirebaseFirestore
) : JobRepository {

 private val jobsCollection = firestore.collection("jobs")
 private val applicationsCollection = firestore.collection("job_applications")
 companion object {
 private const val PAGE_SIZE = 10L
 }

 override suspend fun postJob(job: Job): Result<Unit> {
 return try {

```

```

// Tạo một document mới và lấy ID của nó
val newJobRef = jobsCollection.document()
job.id = newJobRef.id
newJobRef.set(job).await()
Result.success(Unit)
} catch (e: Exception) {
 Result.failure(e)
}
}

override suspend fun getJobs(lastVisibleJob: DocumentSnapshot?):
Result<Pair<List<Job>, DocumentSnapshot?>> {
 return try {
 val query = jobsCollection
 .orderBy("createdAt", Query.Direction.DESCENDING)
 .limit(PAGE_SIZE)

 val finalQuery = if (lastVisibleJob != null) {
 query.startAfter(lastVisibleJob)
 } else {
 query
 }

 val querySnapshot = finalQuery.get().await()
 val jobs = querySnapshot.toObjects(Job::class.java)
 val newLastVisible = querySnapshot.documents.lastOrNull()

 Result.success(Pair(jobs, newLastVisible))
 } catch (e: Exception) {
 Result.failure(e)
 }
}

override suspend fun getJobDetails(jobId: String): Result<Job> {
 return try {
 val document = jobsCollection.document(jobId).get().await()
 val job = document.toObject(Job::class.java)
 if (job != null) {
 Result.success(job)
 } else {
 Result.failure(Exception("Không tìm thấy công việc."))
 }
 } catch (e: Exception) {

```

```

 Result.failure(e)
 }
}

override suspend fun applyForJob(jobId: String, employerId: String, applicantId: String): Result<Unit> {
 return try {
 val newApplicationRef = applicationsCollection.document()
 val application = JobApplication(
 id = newApplicationRef.id,
 jobId = jobId,
 employerId = employerId,
 applicantId = applicantId
)
 newApplicationRef.set(application).await()
 Result.success(Unit)
 } catch (e: Exception) {
 Result.failure(e)
 }
}
}

```

### [app/src/main/java/com/example/antamvieclam/data/repository/impl/UserRepositoryImpl.kt](#)

```

package com.example.antamvieclam.data.repository.impl

import android.content.Context
import android.graphics.Bitmap
import android.graphics.BitmapFactory
import android.net.Uri
import com.cloudinary.android.MediaManager
import com.cloudinary.android.callback.ErrorInfo
import com.cloudinary.android.callback.UploadCallback
import com.example.antamvieclam.data.model.User
import com.example.antamvieclam.data.repository.ProfileResult
import com.example.antamvieclam.data.repository.UserRepository
import com.google.firebase.firestore.FirebaseFirestore
import dagger.hilt.android.qualifiers.ApplicationContext
import kotlinx.coroutines.tasks.await
import java.io.ByteArrayOutputStream
import javax.inject.Inject
import kotlin.coroutines.resume
import kotlin.coroutines.suspendCoroutine

```

```
import kotlin.math.roundToInt

interface UserRepository {
 // Sửa lại hàm để nhận User object và Uri
 suspend fun createUserProfile(user: User, imageUri: Uri?): ProfileResult
 suspend fun getUserProfile(uid: String): User?
 suspend fun doesProfileExist(uid: String): Boolean
}

class UserRepositoryImpl @Inject constructor(
 @ApplicationContext private val context: Context,
 private val firestore: FirebaseFirestore,
 private val mediaManager: MediaManager // Đã được provide từ AppModule
) : UserRepository {

 private val usersCollection = firestore.collection("users")

 override suspend fun getUserProfile(uid: String): User? {
 return try {
 usersCollection.document(uid).get().await().toObject(User::class.java)
 } catch (e: Exception) {
 null
 }
 }

 // THAY ĐỔI TRONG HÀM NÀY
 override suspend fun createUserProfile(user: User, imageUri: Uri?): ProfileResult {
 return try {
 if (imageUri != null) {
 // 1. Nén ảnh từ Uri thành một ByteArray
 val compressedImageData = compressImage(imageUri)
 if (compressedImageData == null) {
 return ProfileResult.Error("Không thể xử lý ảnh được chọn.")
 }

 // 2. Tải lên dữ liệu ảnh đã nén
 val uploadedImageUrl = uploadImage(compressedImageData)
 if (uploadedImageUrl != null) {
 user.profileImageUrl = uploadedImageUrl
 } else {
 return ProfileResult.Error("Tải ảnh lên thất bại.")
 }
 }
 } catch (e: Exception) {
 return ProfileResult.Error("Lỗi khi tạo hồ sơ.")
 }
 }
}
```

```

 }

 usersCollection.document(user.uid).set(user).await()
 ProfileResult.Success

} catch (e: Exception) {
 ProfileResult.Error(e.localizedMessage ?: "Tạo hồ sơ thất bại.")
}

// HÀM MỚI: Để nén ảnh
private fun compressImage(uri: Uri): ByteArray? {
 // Sử dụng ContentResolver để lấy luồng dữ liệu từ Uri
 val inputStream = context.contentResolver.openInputStream(uri) ?: return null

 // 1. Decode luồng dữ liệu thành một đối tượng Bitmap
 val originalBitmap = BitmapFactory.decodeStream(inputStream)

 // 2. Resize ảnh nếu nó quá lớn (ví dụ: giữ chiều lớn nhất là 1080px)
 val maxHeight = 1080.0
 val maxWidth = 1080.0
 val ratio = (originalBitmap.width.toDouble() /
 originalBitmap.height.toDouble()).coerceAtLeast(1.0)

 val newWidth = (maxWidth / ratio).roundToInt()
 val newHeight = (maxHeight / ratio).roundToInt()

 val resizedBitmap = Bitmap.createScaledBitmap(originalBitmap, newWidth, newHeight,
true)

 // 3. Nén ảnh đã resize thành định dạng JPEG với chất lượng 80%
 val outputStream = ByteArrayOutputStream()
 resizedBitmap.compress(Bitmap.CompressFormat.JPEG, 80, outputStream)

 return outputStream.toByteArray()
}

// THAY ĐỔI HÀM NÀY: Giờ sẽ nhận vào ByteArray thay vì Uri
private suspend fun uploadImage(imageData: ByteArray): String? {
 return suspendCoroutine { continuation ->
 // Sử dụng phương thức upload nhận ByteArray
 mediaManager.upload(imageData)
 }
}

```

```

.callback(object : UploadCallback {
 override fun onStart(requestId: String) {}
 override fun onProgress(requestId: String, bytes: Long, totalBytes: Long) {}

 override fun onSuccess(requestId: String, resultData: Map<*, *>) {
 val secureUrl = resultData["secure_url"] as? String
 continuation.resume(secureUrl)
 }

 override fun onError(requestId: String, error: ErrorInfo) {
 continuation.resume(null)
 }

 override fun onReschedule(requestId: String, error: ErrorInfo) {}
}).dispatch()
}
}

override suspend fun doesProfileExist(uid: String): Boolean {
 return try {
 usersCollection.document(uid).get().await().exists()
 } catch (e: Exception) {
 false
 }
}
}

```

### [app/src/main/java/com/example/antamvieclam/di/AppModule.kt](#)

package com.example.antamvieclam.di

```

import com.cloudinary.android.MediaManager
import com.google.firebase.auth.FirebaseAuth
import com.google.firebaseio.firebaseio.FirebaseFirestore
import dagger.Module
import dagger.Provides
import dagger.hilt.InstallIn
import dagger.hilt.components.SingletonComponent
import javax.inject.Singleton

```

@Module

@InstallIn(SingletonComponent::class) // Dependencies sẽ sống sót suốt vòng đời ứng dụng

object AppModule {

```

@Provides
@Singleton // Chỉ tạo một instance duy nhất
fun provideFirebaseAuth(): FirebaseAuth {
 return FirebaseAuth.getInstance()
}

@Provides
@Singleton
fun provideFirestore(): FirebaseFirestore {
 return FirebaseFirestore.getInstance()
}

@Provides
@Singleton
fun provideCloudinary(): MediaManager {
 // Vì MediaManager đã được khởi tạo trong Application class,
 // ở đây chúng ta chỉ cần lấy ra instance đã tồn tại đó.
 return MediaManager.get()
}
}

```

### [app/src/main/java/com/example/antamvieclam/di/RepositoryModule.kt](#)

package com.example.antamvieclam.di

```

import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.JobRepository
import com.example.antamvieclam.data.repository.UserRepository
import com.example.antamvieclam.data.repository.impl.AuthRepositoryImpl
import com.example.antamvieclam.data.repository.impl.JobRepositoryImpl
import com.example.antamvieclam.data.repository.impl.UserRepositoryImpl
import dagger.Binds
import dagger.Module
import dagger.hilt.InstallIn
import dagger.hilt.components.SingletonComponent
import javax.inject.Singleton

@Module
@InstallIn(SingletonComponent::class) // Cài đặt module này vào ApplicationComponent
abstract class RepositoryModule {

 @Binds
 @Singleton // Đảm bảo chỉ có một instance duy nhất của repository trong toàn app
 abstract fun bindAuthRepository(

```

```

 authRepositoryImpl: AuthRepositoryImpl
): AuthRepository // Khi ai đó yêu cầu AuthRepository, Hilt sẽ cung cấp
AuthRepositoryImpl

 @Binds
 @Singleton
 abstract fun bindUserRepository(
 userRepositoryImpl: UserRepositoryImpl
): UserRepository // Tương tự, yêu cầu UserRepository -> cung cấp UserRepositoryImpl

 @Binds
 @Singleton
 abstract fun bindJobRepository(
 jobRepositoryImpl: JobRepositoryImpl
): JobRepository // Khi ai đó yêu cầu JobRepository, Hilt sẽ cung cấp JobRepositoryImpl
}

```

[app/src/main/java/com/example/antamvieclam/ui/auth/AuthViewModel.kt](#)

```
package com.example.antamvieclam.ui.auth
```

```

import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.AuthResult
import com.google.firebase.auth.AuthCredential
import dagger.hilt.android.lifecycle.HiltViewModel
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.asStateFlow
import kotlinx.coroutines.launch
import javax.inject.Inject
import com.example.antamvieclam.data.repository.UserRepository

```

```

sealed class AuthUiState {
 object Idle : AuthUiState()
 object Loading : AuthUiState()
 data class LoginSuccess(val profileExists: Boolean) : AuthUiState()
 data class Error(val message: String) : AuthUiState()
}

```

```

@HiltViewModel
class AuthViewModel @Inject constructor(
 private val authRepository: AuthRepository,
 private val userRepository: UserRepository
)

```

```

) : ViewModel() {

 private val _uiState = MutableStateFlow<AuthUiState>(AuthUiState.Idle)
 val uiState = _uiState.asStateFlow()

 fun signInWithGoogle(credential: AuthCredential) {
 _uiState.value = AuthUiState.Loading
 viewModelScope.launch {
 when (val result = authRepository.signInWithGoogle(credential)) {
 is AuthResult.Success -> {
 // Đăng nhập thành công, giờ kiểm tra profile
 val uid = authRepository.getCurrentUserId()
 if (uid != null) {
 // Dùng userRepository để kiểm tra
 val profileExists = userRepository.doesProfileExist(uid)
 _uiState.value = AuthUiState.LoginSuccess(profileExists)
 } else {
 _uiState.value = AuthUiState.Error("Không lấy được thông tin người dùng.")
 }
 }
 is AuthResult.Error -> {
 _uiState.value = AuthUiState.Error(
 result.exception.localizedMessage ?: "Đăng nhập thất bại."
)
 }
 }
 }
 }

 fun resetState() {
 _uiState.value = AuthUiState.Idle
 }

 fun signOut() {
 authRepository.signOut()
 }
}

app/src/main/java/com/example/antamvieclam/ui/auth/GoogleAuthUiClient.kt
package com.example.antamvieclam.ui.auth

import android.content.Context

```

```
import android.content.Intent
import android.content.IntentSender
import com.example.antamvieclam.R // <- Thêm dòng này
import com.google.android.gms.auth.api.identity.BeginSignInRequest
import com.google.android.gms.auth.api.identity.Identity
import com.google.android.gms.auth.api.identity.SignInClient
import com.google.firebase.auth.GoogleAuthProvider
import kotlinx.coroutines.tasks.await

class GoogleAuthUiClient(
 private val context: Context,
 private val oneTapClient: SignInClient
) {
 suspend fun signIn(): IntentSender? {
 val result = try {
 oneTapClient.beginSignIn(
 BeginSignInRequest.builder()
 .setGoogleIdTokenRequestOptions(
 BeginSignInRequest.GoogleIdTokenRequestOptions.builder()
 .setSupported(true)
 // ID này lấy từ file google-services.json
 // Hoặc vào Google Cloud Console -> APIs & Services -> Credentials
 // -> OAuth 2.0 Client IDs -> Web client (Auto-created by Google Service)
 .setServerClientId(context.getString(R.string.default_web_client_id))
 .setFilterByAuthorizedAccounts(false)
 .build()
)
 .setAutoSelectEnabled(true)
 .build()
).await()
 } catch (e: Exception) {
 e.printStackTrace()
 null
 }
 return result?.pendingIntent?.intentSender
 }

 fun getSignInCredentialFromIntent(intent: Intent) =
 oneTapClient.getSignInCredentialFromIntent(intent).googleIdToken?.let {
 GoogleAuthProvider.getCredential(it, null)
 }
 }
```

```
app/src/main/java/com/example/antamvieclam/ui/auth/HomeScreen.kt
// app/src/main/java/com/example/antamvieclam/ui/auth/HomeScreen.kt
package com.example.antamvieclam.ui.auth
```

```
import androidx.compose.foundation.layout.*
import androidx.compose.material3.CircularProgressIndicator
import androidx.compose.material3.Surface
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.hilt.navigation.compose.hiltViewModel
import androidx.lifecycle.compose.collectAsStateWithLifecycle
import com.example.antamvieclam.data.model.UserType
import com.example.antamvieclam.ui.home.EmployerHomeScreen
import com.example.antamvieclam.ui.home.HomeUiState
import com.example.antamvieclam.ui.home.HomeViewModel
import com.example.antamvieclam.ui.home.WorkerHomeScreen
```

```
@Composable
fun HomeScreen(
 // Các callback điều hướng cần thiết cho các màn hình con
 navigateToLogin: () -> Unit,
 navigateToCreateJob: () -> Unit,
 navigateToJobDetails: (String) -> Unit,
 // ViewModel để lấy thông tin user
 homeViewModel: HomeViewModel = hiltViewModel()
) {
 val uiState by homeViewModel.uiState.collectAsStateWithLifecycle()
```

```
Surface(modifier = Modifier.fillMaxSize()) {
 Box(
 modifier = Modifier.fillMaxSize(),
 contentAlignment = Alignment.Center
) {
 when (val state = uiState) {
 is HomeUiState.Loading -> {
 CircularProgressIndicator()
 }
 is HomeUiState.Error -> {
 Text(text = state.message)
 }
 }
 }
}
```

```
is HomeUiState.Success -> {
 // Dựa vào userType để quyết định hiển thị màn hình nào
 when (state.user.userType) {
 UserType.EMPLOYER -> EmployerHomeScreen(
 navigateToCreateJob = navigateToCreateJob,
 navigateToLogin = navigateToLogin
)
 UserType.WORKER -> WorkerHomeScreen(
 navigateToJobDetails = navigateToJobDetails,
 navigateToLogin = navigateToLogin
)
 }
}
```

app/src/main/java/com/example/antamvieclam/ui/auth/LoginScreen.kt

```
// app/src/main/java/com/example/antamvieclam/ui/auth/LoginScreen.kt
```

```
package com.example.antamvieclam.ui.auth

import android.app.Activity
import android.widget.Toast
import androidx.activity.compose.rememberLauncherForActivityResult
import androidx.activity.result.IntentSenderRequest
import androidx.activity.result.contract.ActivityResultContracts
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.hilt.navigation.compose.hiltViewModel
import androidx.lifecycle.compose.collectAsStateWithLifecycle
```

```
import com.example.antamvieclam.R // Giả sử bạn đã thêm logo google vào drawable
import com.google.android.gms.auth.api.identity.Identity
import kotlinx.coroutines.launch

@Composable
fun LoginScreen(
 viewModel: AuthViewModel = hiltViewModel(),
 navigateToHome: () -> Unit,
 navigateToCreateProfile: () -> Unit
) {
 val uiState by viewModel.uiState.collectAsStateWithLifecycle()
 val context = LocalContext.current
 val coroutineScope = rememberCoroutineScope()

 val googleAuthUiClient by lazy {
 GoogleAuthUiClient(
 context = context,
 oneTapClient = Identity.getSignInClient(context)
)
 }

 val launcher = rememberLauncherForActivityResult(
 contract = ActivityResultContracts.StartIntentSenderForResult(),
 onResult = { result ->
 if (result.resultCode == Activity.RESULT_OK) {
 coroutineScope.launch {
 val credential = googleAuthUiClient.getSignInCredentialFromIntent(result.data ?: return@launch)
 if (credential != null) {
 viewModel.signInWithGoogle(credential)
 }
 }
 }
 }
)

 LaunchedEffect(key1 = uiState) {
 when (val state = uiState) {
 is AuthUiState.LoginSuccess -> {
 Toast.makeText(context, "Đăng nhập thành công!", Toast.LENGTH_SHORT).show()
 if (state.profileExists) {
 navigateToHome()
 } else {

```

```
 navigateToCreateProfile()
 }
 viewModel.resetState() // Reset state sau khi điều hướng
}
is AuthUiState.Error -> {
 Toast.makeText(context, state.message, Toast.LENGTH_LONG).show()
 viewModel.resetState()
}
else -> {}
}

// --- UI DESIGN ---
Surface(modifier = Modifier.fillMaxSize(), color =
MaterialTheme.colorScheme.background) {
 Column(
 modifier = Modifier
 .fillMaxSize()
 .padding(32.dp),
 verticalArrangement = Arrangement.Center,
 horizontalAlignment = Alignment.CenterHorizontally
) {
 // Thay bằng logo của bạn
 Icon(
 painter = painterResource(id = R.drawable.ic_launcher_foreground), // Thay bằng
logo của bạn
 contentDescription = "App Logo",
 modifier = Modifier.size(120.dp),
 tint = MaterialTheme.colorScheme.primary
)
 Spacer(modifier = Modifier.height(16.dp))

 Text(
 text = "Chào mừng đến với\nAn Tâm Việc Làm",
 style = MaterialTheme.typography.headlineSmall,
 textAlign = TextAlign.Center,
 fontWeight = FontWeight.Bold
)

 Text(
 text = "Nền tảng kết nối người lao động tự do",
 style = MaterialTheme.typography.bodyMedium,
 textAlign = TextAlign.Center,
```

```
 color = Color.Gray
)

Spacer(modifier = Modifier.height(64.dp))

GoogleSignInButton(
 isLoading = uiState == AuthUiState.Loading,
 onClick = {
 coroutineScope.launch {
 val signInIntentSender = googleAuthUiClient.signIn()
 launcher.launch(
 IntentSenderRequest.Builder(
 signInIntentSender ?: return@launch
).build()
)
 }
 }
)
}
```

```
@Composable
fun GoogleSignInButton(
 isLoading: Boolean = false,
 onClick: () -> Unit
) {
 Button(
 onClick = onClick,
 enabled = !isLoading,
 modifier = Modifier
 .fillMaxWidth()
 .height(50.dp),
 colors = ButtonDefaults.buttonColors(
 containerColor = Color.White,
 contentColor = Color.Black
),
 elevation = ButtonDefaults.buttonElevation(defaultElevation = 2.dp)
) {
 if (isLoading) {
 CircularProgressIndicator(
 modifier = Modifier.size(24.dp),
 color = MaterialTheme.colorScheme.primary,

```

```

 strokeWidth = 2.dp
)
} else {
 Row(
 verticalAlignment = Alignment.CenterVertically,
 horizontalArrangement = Arrangement.Center
){
 // Thêm file ảnh logo google vào thư mục res/drawable
 Image(
 painter = painterResource(id = R.drawable.ic_google_logo),
 contentDescription = "Google logo",
 modifier = Modifier.size(24.dp)
)
 Spacer(modifier = Modifier.width(12.dp))
 Text("Đăng nhập với Google", fontWeight = FontWeight.SemiBold)
 }
}
}
}
}

```

### [app/src/main/java/com/example/antamvieclam/ui/home/EmployerHomeScreen.kt](#)

```
// app/src/main/java/com/example/antamvieclam/ui/home/EmployerHomeScreen.kt
package com.example.antamvieclam.ui.home
```

```

import androidx.compose.foundation.layout.*
import androidx.compose.material.icons(Icons
import androidx.compose.material.icons.filled.Add
import androidx.compose.material.icons.filled.Logout // <-- THÊM IMPORT
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.hilt.navigation.compose.hiltViewModel
import com.example.antamvieclam.ui.auth.AuthViewModel

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun EmployerHomeScreen(
 navigateToCreateJob: () -> Unit,

```

```
 navigateToLogin: () -> Unit,
 authViewModel: AuthViewModel = hiltViewModel()
) {
Scaffold(
 topBar = {
 TopAppBar(
 title = { Text("Quản Lý Tin Tuyển Dụng") },
 // --- THÊM KHỐI NÀY VÀO ---
 actions = {
 IconButton(onClick = {
 authViewModel.signOut()
 navigateToLogin()
 }) {
 Icon(
 imageVector = Icons.Default.Logout,
 contentDescription = "Đăng xuất"
)
 }
 },
 colors = TopAppBarDefaults.topAppBarColors(
 containerColor = MaterialTheme.colorScheme.primary,
 titleContentColor = Color.White,
 actionIconContentColor = Color.White // Quan trọng: Đặt màu cho icon
)
)
 },
 floatingActionButton = {
 FloatingActionButton(onClick = navigateToCreateJob) {
 Icon(Icons.Default.Add, contentDescription = "Đăng tin mới")
 }
 }
) { paddingValues ->
 Box(
 modifier = Modifier
 .fillMaxSize()
 .padding(paddingValues)
 .padding(16.dp),
 contentAlignment = Alignment.Center
) {
 Text(
 text = "Bạn chưa đăng tin tuyển dụng nào.\nNhấn nút '+' để bắt đầu.",
 textAlign = TextAlign.Center,
 style = MaterialTheme.typography.bodyLarge,
```

```

 color = Color.Gray
)
}
}
}

app/src/main/java/com/example/antamvieclam/ui/home/HomeViewModel.kt
// app/src/main/java/com/example/antamvieclam/ui/home/HomeViewModel.kt
package com.example.antamvieclam.ui.home

import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.antamvieclam.data.model.User
import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.UserRepository
import dagger.hilt.android.lifecycle.HiltViewModel
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.asStateFlow
import kotlinx.coroutines.launch
import javax.inject.Inject

// Lớp trạng thái cho màn hình Home
sealed class HomeUiState {
 object Loading : HomeUiState()
 data class Success(val user: User) : HomeUiState()
 data class Error(val message: String) : HomeUiState()
}

@HiltViewModel
class HomeViewModel @Inject constructor(
 private val authRepository: AuthRepository,
 private val userRepository: UserRepository
) : ViewModel() {

 private val _uiState = MutableStateFlow<HomeUiState>(HomeUiState.Loading)
 val uiState = _uiState.asStateFlow()

 init {
 // Tải thông tin người dùng ngay khi ViewModel được tạo
 loadCurrentUser()
 }

 private fun loadCurrentUser() {

```

```

viewModelScope.launch {
 _uiState.value = HomeUiState.Loading
 val userId = authRepository.getCurrentUserId()

 if (userId == null) {
 _uiState.value = HomeUiState.Error("Không thể xác thực người dùng.")
 return@launch
 }

 val user = userRepository.getUserProfile(userId)
 if (user != null) {
 _uiState.value = HomeUiState.Success(user)
 } else {
 // Trường hợp hiếm gặp: đã đăng nhập nhưng không có profile
 _uiState.value = HomeUiState.Error("Không tìm thấy hồ sơ người dùng.")
 }
}
}
}

```

### [`app/src/main/java/com/example/antamvieclam/ui/home/WorkerHomeScreen.kt`](#)

```
// app/src/main/java/com/example/antamvieclam/ui/home/WorkerHomeScreen.kt
package com.example.antamvieclam.ui.home
```

```

import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.material.icons(Icons
import androidx.compose.material.icons.filled.Logout
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.runtime.remember
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.hilt.navigation.compose.hiltViewModel
import com.example.antamvieclam.data.model.Job
import com.example.antamvieclam.ui.auth.AuthViewModel

```

```

@OptIn(ExperimentalMaterial3Api::class)
@Composable

```

```
fun WorkerHomeScreen(
 navigateToJobDetails: (String) -> Unit,
 navigateToLogin: () -> Unit,
 authViewModel: AuthViewModel = hiltViewModel()
) {
 // Dữ liệu giả để hiển thị UI
 val jobs = remember {
 listOf(
 Job(id = "1", title = "Phụ hồ công trình Quận 7", payRate = 300000.0, addressString =
 "Quận 7, TP. HCM"),
 Job(id = "2", title = "Bốc vác kho hàng Giaohangtietkiem", payRate = 25000.0,
 addressString = "Quận 12, TP. HCM"),
 Job(id = "3", title = "Giao hàng bán thời gian", payRate = 28000.0, addressString =
 "Bình Thạnh, TP. HCM"),
)
 }

 Scaffold(
 topBar = {
 TopAppBar(
 title = { Text("Tìm Việc Quanh Đây") },
 actions = {
 IconButton(onClick = {
 authViewModel.signOut()
 navigateToLogin()
 }) {
 Icon(Icons.Default.Logout, contentDescription = "Đăng xuất")
 }
 },
 colors = TopAppBarDefaults.topAppBarColors(
 containerColor = MaterialTheme.colorScheme.primary,
 titleContentColor = Color.White,
 actionIconContentColor = Color.White
)
)
 }
) { paddingValues ->
 LazyColumn(
 modifier = Modifier.fillMaxSize().padding(paddingValues),
 contentPadding = PaddingValues(16.dp),
 verticalArrangement = Arrangement.spacedBy(12.dp)
) {
 items(jobs.size) { index ->
```

```

 JobItemCard(job = jobs[index], onJobClick = navigateToJobDetails)
 }
}
}
}

@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun JobItemCard(
 job: Job,
 onJobClick: (String) -> Unit
) {
 Card(
 modifier = Modifier.fillMaxWidth().clickable { onJobClick(job.id) },
 elevation = CardDefaults.cardElevation(defaultElevation = 2.dp),
 colors = CardDefaults.cardColors(containerColor =
MaterialTheme.colorScheme.surface) // Sử dụng màu nền của theme
) {
 Column(modifier = Modifier.padding(16.dp)) {
 Text(
 text = job.title,
 style = MaterialTheme.typography.titleMedium,
 fontWeight = FontWeight.Bold
)
 Spacer(modifier = Modifier.height(4.dp))
 Text(
 text = job.addressString,
 style = MaterialTheme.typography.bodyMedium,
 color = MaterialTheme.colorScheme.onSurfaceVariant
)
 Spacer(modifier = Modifier.height(8.dp))
 Text(
 text = "${job.payRate.toLong()} VNĐ / giờ", // Cần format payType sau
 style = MaterialTheme.typography.bodyLarge,
 color = MaterialTheme.colorScheme.primary,
 fontWeight = FontWeight.SemiBold
)
 }
 }
}

```

## [app/src/main/java/com/example/antamvieclam/ui/job\\_details/JobDetailsScreen.kt](#)

```
// app/src/main/java/com/example/antamvieclam/ui/job_details/JobDetailsScreen.kt
package com.example.antamvieclam.ui.job_details
```

```
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons(Icons)
import androidx.compose.material.icons.filled.ArrowBack
import androidx.compose.material3.*
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
```

```
@OptIn(ExperimentalMaterial3Api::class)
@Composable
fun JobDetailsScreen(
 jobId: String?,
 onNavigateBack: () -> Unit
) {
 // TODO: Kết nối ViewModel để tải chi tiết công việc bằng jobId
```

```
Scaffold(
 topBar = {
 TopAppBar(
 title = { Text("Chi Tiết Công Việc") },
 navigationIcon = {
 IconButton(onClick = onNavigateBack) {
 Icon(Icons.Default.ArrowBack, contentDescription = "Quay lại")
 }
 }
),
 bottomBar = {
 Button(
 onClick = { /* TODO: Gọi viewModel.applyForJob() */ },
 modifier = Modifier
 .fillMaxWidth()
 .padding(16.dp)
 .height(50.dp)
)
 }
 }
)
```

```

) {
 Text("ỨNG TUYỂN NGAY", fontWeight = FontWeight.Bold)
 }
 }
) { paddingValues ->
 Column(
 modifier = Modifier
 .fillMaxSize()
 .padding(paddingValues)
 .padding(16.dp)
 .verticalScroll(rememberScrollState()),
 verticalArrangement = Arrangement.spacedBy(16.dp)
) {
 // Đây là dữ liệu giả, sẽ được thay thế bằng dữ liệu thật từ ViewModel
 Text(
 text = "Phụ hồ công trình Q7",
 style = MaterialTheme.typography.headlineSmall,
 fontWeight = FontWeight.Bold
)
 Divider()
 JobDetailRow(title = "Nhà tuyển dụng", content = "Anh Bảy")
 JobDetailRow(title = "Mức lương", content = "300,000 VNĐ / ngày")
 JobDetailRow(title = "Địa chỉ", content = "123 Nguyễn Thị Thập, P. Tân Phong, Quận 7, TP.HCM")
 Text("Mô tả công việc", style = MaterialTheme.typography.titleMedium, fontWeight =
FontWeight.SemiBold)
 Text(
 text = "Cần người phụ hồ, biết trộn vữa, siêng năng, chịu khó. Làm việc từ 8h sáng
đến 5h chiều, có nghỉ trưa.",
 style = MaterialTheme.typography.bodyLarge
)
 }
}
}

{@Composable
fun JobDetailRow(title: String, content: String) {
 Column {
 Text(text = title, style = MaterialTheme.typography.labelMedium, color = Color.Gray)
 Text(text = content, style = MaterialTheme.typography.bodyLarge)
 }
}
}

```

## [app/src/main/java/com/example/antamvieclam/ui/navigation/AppNavigation.kt](#)

```
// app/src/main/java/com/example/antamvieclam/ui/navigation/AppNavigation.kt
```

```
package com.example.antamvieclam.ui.navigation
```

```
import androidx.compose.runtime.Composable
import androidx.navigation.NavType
import androidx.navigation.compose.NavHost
import androidx.navigation.compose.composable
import androidx.navigation.compose.rememberNavController
import androidx.navigation.navArgument
import com.example.antamvieclam.ui.auth.HomeScreen
import com.example.antamvieclam.ui.auth.LoginScreen
import com.example.antamvieclam.ui.job_details.JobDetailsScreen
import com.example.antamvieclam.ui.posting.CreateJobScreen
import com.example.antamvieclam.ui.profile.CreateProfileScreen
import com.google.firebase.auth.FirebaseAuth
```

```
// Xóa hàm HomeScreen() giả ở đây nếu có
```

```
@Composable
fun AppNavigation() {
 val navController = rememberNavController()
```

```
// SỬA LẠI LOGIC: Nếu đã đăng nhập, vào thẳng HOME_SCREEN
val startDestination = if (FirebaseAuth.getInstance().currentUser != null) {
 Routes.HOME_SCREEN
} else {
 Routes.LOGIN_SCREEN
}
```

```
NavHost(
 navController = navController,
 startDestination = startDestination
) {
 composable(Routes.LOGIN_SCREEN) {
 LoginScreen(
 navigateToHome = {
 navController.navigate(Routes.HOME_SCREEN) {
 popUpTo(Routes.LOGIN_SCREEN) { inclusive = true }
 }
 },
),
 }
}
```

```

 navigateToCreateProfile = {
 navController.navigate(Routes.CREATE_PROFILE_SCREEN) {
 popUpTo(Routes.LOGIN_SCREEN) { inclusive = true }
 }
 }
 }

composable(Routes.CREATE_PROFILE_SCREEN) {
 CreateProfileScreen(
 onProfileCreated = {
 navController.navigate(Routes.HOME_SCREEN) {
 popUpTo(Routes.CREATE_PROFILE_SCREEN) { inclusive = true }
 }
 }
)
}

// SỬA LẠI HOÀN TOÀN KHỐI NÀY
composable(Routes.HOME_SCREEN) {
 HomeScreen(
 navigateToLogin = {
 navController.navigate(Routes.LOGIN_SCREEN) {
 // Xóa hết back stack
 popUpTo(navController.graph.startDestinationId) { inclusive = true }
 }
 },
 // Đã thay thế TODO() bằng hành động điều hướng thật
 navigateToCreateJob = {
 navController.navigate(Routes.CREATE_JOB_SCREEN)
 },
 // Đã thay thế TODO() bằng hành động điều hướng thật, có truyền tham số
 navigateToJobDetails = { jobId ->
 // Tạo route cụ thể cho công việc được chọn, ví dụ: "job_details/abc-123"
 navController.navigate("${Routes.JOB_DETAILS_SCREEN}/$jobId")
 }
 // Không cần truyền viewModel vào đây vì HomeScreen sẽ tự lấy bằng
 hiltViewModel()
)
}

// THÊM COMPOSABLE CHO MÀN HÌNH TẠO VIỆC
composable(Routes.CREATE_JOB_SCREEN) {

```

```

CreateJobScreen(
 onJobPosted = {
 // Sau khi đăng tin thành công, quay lại màn hình trước đó
 navController.popBackStack()
 }
)
}

// THÊM COMPOSABLE CHO MÀN HÌNH CHI TIẾT CÔNG VIỆC
composable(
 route = "${Routes.JOB_DETAILS_SCREEN}/{jobId}", // Định nghĩa route với tham số
"jobId"
 arguments = listOf(navArgument("jobId") { type = NavType.StringType })
) { backStackEntry ->
 // Lấy jobId từ arguments
 JobDetailsScreen(
 jobId = backStackEntry.arguments?.getString("jobId"),
 onNavigateBack = {
 navController.popBackStack()
 }
)
}
}
}

```

### [app/src/main/java/com/example/antamvieclam/ui/navigation/Routes.kt](#)

```
package com.example.antamvieclam.ui.navigation
```

```

object Routes {
 const val LOGIN_SCREEN = "login"
 const val OTP_SCREEN = "otp/{verificationId}" // Chúng ta cần truyền verificationId
 const val CREATE_PROFILE_SCREEN = "create_profile"
 const val HOME_SCREEN = "home" // Màn hình chính sau khi đăng nhập và có hồ sơ
 const val CREATE_JOB_SCREEN = "create_job"
 const val JOB_DETAILS_SCREEN = "job_details"
}

```

### [app/src/main/java/com/example/antamvieclam/ui/posting/CreateJobScreen.kt](#)

```
// app/src/main/java/com/example/antamvieclam/ui/posting/CreateJobScreen.kt
package com.example.antamvieclam.ui.posting
```

```
import android.os.Bundle
```

```
import android.widget.Toast
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons(Icons)
import androidx.compose.material.icons.filled.LocationOn
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.platform.LocalLifecycleOwner
import androidx.compose.ui.unit.dp
import androidx.compose.ui.viewinterop.AndroidView
import androidx.hilt.navigation.compose.hiltViewModel
import androidx.lifecycle.Lifecycle
import androidx.lifecycle.LifecycleEventObserver
import androidx.lifecycle.compose.collectAsStateWithLifecycle
import com.example.antamvieclam.data.model.PayType
import com.google.firebase.firestore.GeoPoint
import com.trackasia.android.TrackAsia
import com.trackasia.android.camera.CameraPosition
import com.trackasia.android.geometry.LatLng
import com.trackasia.android.maps.MapView
import com.trackasia.android.maps.Style

@Composable
fun CreateJobScreen(
 onJobPosted: () -> Unit,
 viewModel: CreateJobViewModel = hiltViewModel()
) {
 val uiState by viewModel.uiState.collectAsStateWithLifecycle()
 val context = LocalContext.current

 var title by remember { mutableStateOf("") }
 var description by remember { mutableStateOf("") }
 var payRate by remember { mutableStateOf("") }
 var address by remember { mutableStateOf("") }
 var payType by remember { mutableStateOf(PayType.PER_HOUR) }

 // State để lưu vị trí trung tâm bản đồ hiện tại
```

```

val defaultLocation = LatLng(10.7769, 106.7009) // TP.HCM
var selectedLatLng by remember { mutableStateOf(defaultLocation) }

LaunchedEffect(uiState) {
 when(val state = uiState) {
 is CreateJobState.Success -> {
 Toast.makeText(context, "Đăng tin thành công!", Toast.LENGTH_SHORT).show()
 onJobPosted()
 }
 is CreateJobState.Error -> {
 Toast.makeText(context, state.message, Toast.LENGTH_LONG).show()
 }
 else -> {}
 }
}

Scaffold(
 topBar = { /* ... TopAppBar ... */ }
) { paddingValues ->
 Column(
 modifier = Modifier
 .fillMaxSize()
 .padding(paddingValues)
 .padding(16.dp)
 .verticalScroll(rememberScrollState())
) {
 Text("Đăng tin tuyển dụng mới", style = MaterialTheme.typography.headlineSmall)
 Spacer(modifier = Modifier.height(16.dp))

 // ... Các OutlinedTextField cho title, description, payRate, address ...
 OutlinedTextField(value = title, onValueChange = {title = it}, label={Text("Tiêu đề
công việc")}, modifier=Modifier.fillMaxWidth())
 Spacer(Modifier.height(8.dp))
 OutlinedTextField(value = description, onValueChange = {description = it},
 label={Text("Mô tả chi tiết")}, modifier=Modifier.fillMaxWidth())
 Spacer(Modifier.height(8.dp))
 OutlinedTextField(value = payRate, onValueChange = {payRate = it},
 label={Text("Mức lương (VD: 50000)"")}, modifier=Modifier.fillMaxWidth())
 Spacer(Modifier.height(8.dp))
 OutlinedTextField(value = address, onValueChange = {address = it}, label={Text("Địa
chỉ cụ thể")}, modifier=Modifier.fillMaxWidth())
 Spacer(Modifier.height(16.dp))
 }
}

```

```

// --- Vùng bản đồ TrackAsia ---
Text("Ghim vị trí công việc", style = MaterialTheme.typography.titleMedium)
Spacer(modifier = Modifier.height(8.dp))
Box(
 modifier = Modifier
 .fillMaxWidth()
 .height(300.dp)
 .clip(MaterialTheme.shapes.medium),
 contentAlignment = Alignment.Center
) {
 TrackAsiaMapPicker(
 initialPosition = defaultLocation,
 onCameraMove = { newLatLng ->
 // Cập nhật state khi người dùng di chuyển bản đồ
 selectedLatLng = newLatLng
 }
)
 // Marker cố định ở giữa
 Icon(
 imageVector = Icons.Default.LocationOn,
 contentDescription = "Pin",
 tint = Color.Red,
 modifier = Modifier.size(40.dp).padding(bottom = 20.dp)
)
}

Spacer(modifier = Modifier.height(24.dp))

Button(
 onClick = {
 val geoPoint = GeoPoint(selectedLatLng.latitude, selectedLatLng.longitude)
 viewModel.postJob(title, description, payRate, payType, address, geoPoint)
 },
 enabled = uiState != CreateJobState.Loading,
 modifier = Modifier.fillMaxWidth().height(50.dp)
) {
 if (uiState == CreateJobState.Loading) {
 CircularProgressIndicator(modifier = Modifier.size(24.dp))
 } else {
 Text("ĐĂNG TIN")
 }
}

```

```
 }
 }

 @Composable
 fun TrackAsiaMapPicker(
 initialPosition: LatLng,
 onCameraMove: (LatLng) -> Unit
) {
 val context = LocalContext.current
 // Tạo và nhớ instance của MapView
 val mapView = remember {
 MapView(context).apply {
 // Khởi tạo TrackAsia instance trước khi sử dụng MapView
 MapView(context)
 }
 }

 // Quản lý vòng đời của MapView một cách an toàn
 val lifecycle = LocalLifecycleOwner.current.lifecycle
 DisposableEffect(lifecycle, mapView) {
 val lifecycleObserver = LifecycleEventObserver { _, event ->
 when (event) {
 Lifecycle.Event.ON_CREATE -> mapView.onCreate(Bundle())
 Lifecycle.Event.ON_START -> mapView.onStart()
 Lifecycle.Event.ON_RESUME -> mapView.onResume()
 Lifecycle.Event.ON_PAUSE -> mapView.onPause()
 Lifecycle.Event.ON_STOP -> mapView.onStop()
 Lifecycle.Event.ON_DESTROY -> mapView.onDestroy()
 else -> {}
 }
 }
 lifecycle.addObserver(lifecycleObserver)
 onDispose {
 lifecycle.removeObserver(lifecycleObserver)
 }
 }

 // Sử dụng AndroidView để hiển thị MapView
 AndroidView(
 factory = {
 mapView.apply {
 getMapAsync { trackAsiaMap ->
 // TODO: THAY "public_key" BẰNG API KEY CỦA BẠN
 }
 }
 }
)
 }
}
```

```
val styleUrl = "https://maps.track-
asia.com/styles/v1/streets.json?key=52fdb6b306931761836057e5580a05be7"
trackAsiaMap.setStyle(Style.Builder().fromUri(styleUrl))

trackAsiaMap.cameraPosition = CameraPosition.Builder()
 .target(initialPosition)
 .zoom(15.0)
 .build()

// Lắng nghe sự kiện khi người dùng ngừng di chuyển bản đồ
trackAsiaMap.addOnCameraIdleListener {
 // Cập nhật tọa độ mới lên Composable state
 trackAsiaMap.cameraPosition.target?.let { p1 -> onCameraMove(p1) }
}
}
}
}
}
}
```

app/src/main/java/com/example/antamvieclam/ui/posting/CreateJobViewModel.kt

```
// app/src/main/java/com/example/antamvieclam/ui/posting/CreateJobViewModel.kt
package com.example.antamvieclam.ui.posting
```

```
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.antamvieclam.data.model.Job
import com.example.antamvieclam.data.model.PayType
import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.JobRepository
import com.example.antamvieclam.data.repository.UserRepository
import com.google.firebase.firestore.GeoPoint
import dagger.hilt.android.lifecycle.HiltViewModel
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.asStateFlow
import kotlinx.coroutines.launch
import javax.inject.Inject

sealed class CreateJobState {
 object Idle : CreateJobState()
 object Loading : CreateJobState()
 object Success : CreateJobState()
```

```
data class Error(val message: String) : CreateJobState()
}

@HiltViewModel
class CreateJobViewModel @Inject constructor(
 private val jobRepository: JobRepository,
 private val authRepository: AuthRepository,
 private val userRepository: UserRepository
) : ViewModel() {

 private val _uiState = MutableStateFlow<CreateJobState>(CreateJobState.Idle)
 val uiState = _uiState.asStateFlow()

 fun postJob(
 title: String,
 description: String,
 payRate: String,
 payType: PayType,
 address: String,
 location: GeoPoint?
) {
 _uiState.value = CreateJobState.Loading

 // --- Validation ---
 if (title.isBlank() || description.isBlank() || payRate.isBlank() || address.isBlank()) {
 _uiState.value = CreateJobState.Error("Vui lòng điền đầy đủ thông tin.")
 return
 }
 if (location == null) {
 _uiState.value = CreateJobState.Error("Vui lòng chọn vị trí trên bản đồ.")
 return
 }
 val payRateDouble = payRate.toDoubleOrNull()
 if (payRateDouble == null || payRateDouble <= 0) {
 _uiState.value = CreateJobState.Error("Mức lương không hợp lệ.")
 return
 }

 viewModelScope.launch {
 val userId = authRepository.getCurrentUserId()
 if (userId == null) {
 _uiState.value = CreateJobState.Error("Không thể xác thực người dùng.")
 return@launch
 }
 }
 }
}
```

```

 }

 // Lấy thông tin NTD để nhúng vào job object
 val employer = userRepository.getUserProfile(userId)
 if (employer == null) {
 _uiState.value = CreateJobState.Error("Không tìm thấy hồ sơ nhà tuyển dụng.")
 return@launch
 }

 val newJob = Job(
 employerId = userId,
 employerName = employer.fullName,
 employerProfileUrl = employer.profileImageUrl,
 title = title,
 description = description,
 payRate = payRateDouble,
 payType = payType,
 addressString = address,
 location = location
)

 jobRepository.postJob(newJob)
 .onSuccess { _uiState.value = CreateJobState.Success }
 .onFailure { _uiState.value = CreateJobState.Error(it.message ?: "Đã xảy ra lỗi") }
 }
}
}

```

### [app/src/main/java/com/example/antamvieclam/ui/profile/CreateProfileScreen.kt](#)

```
// app/src/main/java/com/example/antamvieclam/ui/profile/CreateProfileScreen.kt
```

```
package com.example.antamvieclam.ui.profile
```

```

import android.net.Uri
import android.widget.Toast
import androidx.activity.compose.rememberLauncherForActivityResult
import androidx.activity.result.contract.ActivityResultContracts
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.border
import androidx.compose.foundation.clickable
import androidx.compose.foundation.layout.*

```

```
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.shape.CircleShape
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons(Icons)
import androidx.compose.material.icons.filled.Edit
import androidx.compose.material.icons.filled.Person
import androidx.compose.material3.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.clip
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.hilt.navigation.compose.hiltViewModel
import androidx.lifecycle.compose.collectAsStateWithLifecycle
import coil.compose.rememberAsyncImagePainter
import com.example.antamvieclam.data.model.UserType
import com.google.firebase.auth.FirebaseAuth

@Composable
fun CreateProfileScreen(
 viewModel: ProfileViewModel = hiltViewModel(),
 onProfileCreated: () -> Unit
) {
 val uiState by viewModel.uiState.collectAsStateWithLifecycle()
 val context = LocalContext.current

 val suggestedName = FirebaseAuth.getInstance().currentUser?.displayName ?: ""
 var fullName by remember { mutableStateOf(suggestedName) }
 var selectedUserType by remember { mutableStateOf(UserType.WORKER) }
 var imageUri by remember { mutableStateOf<Uri?>(null) }

 val imagePickerLauncher = rememberLauncherForActivityResult(
 contract = ActivityResultContracts.GetContent()
) { uri: Uri? ->
 imageUri = uri
 }

 LaunchedEffect(key1 = uiState) {
```

```

when (val state = uiState) {
 is ProfileUiState.SaveSuccess -> {
 Toast.makeText(context, "Tạo hồ sơ thành công!", Toast.LENGTH_SHORT).show()
 onProfileCreated()
 }
 is ProfileUiState.Error -> {
 Toast.makeText(context, state.message, Toast.LENGTH_LONG).show()
 }
 else -> {}
}

// --- UI DESIGN ---
Surface(modifier = Modifier.fillMaxSize()) {
 Column(
 modifier = Modifier
 .fillMaxSize()
 .padding(24.dp)
 .verticalScroll(rememberScrollState()), // Cho phép cuộn khi bàn phím hiện
 horizontalAlignment = Alignment.CenterHorizontally
) {
 Spacer(modifier = Modifier.height(32.dp))
 Text("Tạo Hồ Sơ Của Bạn", style = MaterialTheme.typography.headlineMedium,
 fontWeight = FontWeight.Bold)
 Text("Hãy cho chúng tôi biết thêm về bạn", style =
 MaterialTheme.typography.bodyMedium, color = Color.Gray)
 Spacer(modifier = Modifier.height(32.dp))

 // Vùng chọn ảnh đại diện cài tiến
 Box(contentAlignment = Alignment.BottomEnd) {
 Image(
 painter = rememberAsyncImagePainter(
 model = imageUri ?: "https://via.placeholder.com/150"
),
 contentDescription = "Ảnh đại diện",
 modifier = Modifier
 .size(120.dp)
 .clip(CircleShape)
 .border(2.dp, MaterialTheme.colorScheme.primary, CircleShape)
 .clickable { imagePickerLauncher.launch("image/*") },
 contentScale = ContentScale.Crop
)
 Box(

```

```
 modifier = Modifier
 .size(36.dp)
 .clip(CircleShape)
 .background(MaterialTheme.colorScheme.primary)
 .padding(6.dp)
) {
 Icon(
 imageVector = Icons.Default.Edit,
 contentDescription = "Chỉnh sửa ảnh",
 tint = Color.White
)
}
}

Spacer(modifier = Modifier.height(32.dp))

// Trường nhập Họ và Tên
OutlinedTextField(
 value = fullName,
 onValueChange = { fullName = it },
 label = { Text("Họ và Tên") },
 modifier = Modifier.fillMaxWidth(),
 singleLine = true,
 leadingIcon = { Icon(Icons.Default.Person, contentDescription = null)}
)

Spacer(modifier = Modifier.height(24.dp))

// Vùng chọn vai trò cải tiến
Text("Bạn là:", modifier = Modifier.fillMaxWidth(), fontWeight =
FontWeight.SemiBold)
Spacer(modifier = Modifier.height(8.dp))
Column(modifier = Modifier.fillMaxWidth()) {
 RoleSelectionRow(
 text = "Người lao động",
 selected = selectedUserType == UserType.WORKER,
 onClick = { selectedUserType = UserType.WORKER }
)
 Spacer(modifier = Modifier.height(8.dp))
 RoleSelectionRow(
 text = "Nhà tuyển dụng",
 selected = selectedUserType == UserType.EMPLOYER,
 onClick = { selectedUserType = UserType.EMPLOYER }
)
}
```

```
)
 }

 Spacer(modifier = Modifier.weight(1f)) // Đẩy nút xuống dưới

 // Nút Lưu
 Button(
 onClick = {
 viewModel.saveUserProfile(fullName, selectedUserType, imageUri)
 },
 modifier = Modifier
 .fillMaxWidth()
 .height(50.dp),
 enabled = uiState != ProfileUiState.Loading
) {
 if (uiState == ProfileUiState.Loading) {
 CircularProgressIndicator(
 modifier = Modifier.size(24.dp),
 color = MaterialTheme.colorScheme.onPrimary
)
 } else {
 Text("HOÀN TẤT", fontSize = 16.sp, fontWeight = FontWeight.Bold)
 }
 }
}
```

```
@Composable
fun RoleSelectionRow(text: String, selected: Boolean, onClick: () -> Unit) {
 Row(
 modifier = Modifier
 .fillMaxWidth()
 .border(
 width = 1.dp,
 color = if (selected) MaterialTheme.colorScheme.primary else Color.LightGray,
 shape = MaterialTheme.shapes.medium
)
 .clip(MaterialTheme.shapes.medium)
 .clickable(onClick = onClick)
 .padding(horizontal = 16.dp, vertical = 12.dp),
 verticalAlignment = Alignment.CenterVertically
)
```

```

) {
 RadioButton(
 selected = selected,
 onClick = onClick
)
 Spacer(modifier = Modifier.width(8.dp))
 Text(text = text, style = MaterialTheme.typography.bodyLarge)
}
}

app/src/main/java/com/example/antamvieclam/ui/profile/ProfileViewModel.kt
package com.example.antamvieclam.ui.profile

import android.net.Uri
import androidx.lifecycle.ViewModel
import androidx.lifecycle.viewModelScope
import com.example.antamvieclam.data.model.User
import com.example.antamvieclam.data.model.UserType
import com.example.antamvieclam.data.repository.AuthRepository
import com.example.antamvieclam.data.repository.ProfileResult
import com.example.antamvieclam.data.repository.UserRepository
import com.google.firebase.auth.FirebaseAuth
import dagger.hilt.android.lifecycle.HiltViewModel
import kotlinx.coroutines.flow.MutableStateFlow
import kotlinx.coroutines.flow.asStateFlow
import kotlinx.coroutines.launch
import javax.inject.Inject

sealed class ProfileUiState {
 object Idle : ProfileUiState()
 object Loading : ProfileUiState()
 object SaveSuccess : ProfileUiState()
 data class Error(val message: String) : ProfileUiState()
}

@HiltViewModel
class ProfileViewModel @Inject constructor(
 private val userRepository: UserRepository,
 private val authRepository: AuthRepository
) : ViewModel() {

 private val _uiState = MutableStateFlow<ProfileUiState>(ProfileUiState.Idle)

```

```

val uiState = _uiState.asStateFlow()

fun saveUserProfile(fullName: String, userType: UserType, imageUri: Uri?) {
 _uiState.value = ProfileUiState.Loading

 if (fullName.isBlank()) {
 _uiState.value = ProfileUiState.Error("Vui lòng nhập họ và tên.")
 return
 }

 val currentUserId = authRepository.getCurrentUserId()
 if (currentUserId == null) {
 _uiState.value = ProfileUiState.Error("Người dùng chưa đăng nhập.")
 return
 }

 viewModelScope.launch {
 val user = User(
 uid = currentUserId,
 phoneNumber = FirebaseAuth.getInstance().currentUser?.phoneNumber,
 fullName = fullName,
 userType = userType
)

 val result = userRepository.createUserProfile(user, imageUri)
 when (result) {
 is ProfileResult.Success -> {
 _uiState.value = ProfileUiState.SaveSuccess
 }
 is ProfileResult.Error -> {
 _uiState.value = ProfileUiState.Error(result.message)
 }
 }
 }
}

```

[app/src/main/java/com/example/antamvieclam/ui/theme/Color.kt](#)

package com.example.antamvieclam.ui.theme

import androidx.compose.ui.graphics.Color

val Purple80 = Color(0xFFD0BCFF)

```
val PurpleGrey80 = Color(0xFFCCC2DC)
val Pink80 = Color(0xFFEFB8C8)
```

```
val Purple40 = Color(0xFF6650a4)
val PurpleGrey40 = Color(0xFF625b71)
val Pink40 = Color(0xFF7D5260)
```

[app/src/main/java/com/example/antamvieclam/ui/theme/Theme.kt](#)  
package com.example.antamvieclam.ui.theme

```
import android.app.Activity
import android.os.Build
import androidx.compose.foundation.isSystemInDarkTheme
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.darkColorScheme
import androidx.compose.material3.dynamicDarkColorScheme
import androidx.compose.material3.dynamicLightColorScheme
import androidx.compose.material3.lightColorScheme
import androidx.compose.runtime.Composable
import androidx.compose.ui.platform.LocalContext
```

```
private val DarkColorScheme = darkColorScheme(
 primary = Purple80,
 secondary = PurpleGrey80,
 tertiary = Pink80
)
```

```
private val LightColorScheme = lightColorScheme(
 primary = Purple40,
 secondary = PurpleGrey40,
 tertiary = Pink40
```

```
/* Other default colors to override
background = Color(0xFFFFFBFE),
surface = Color(0xFFFFFBFE),
onPrimary = Color.White,
onSecondary = Color.White,
onTertiary = Color.White,
onBackground = Color(0xFF1C1B1F),
onSurface = Color(0xFF1C1B1F),
*/
)
```

```

@Composable
fun AnTamViecLamTheme(
 darkTheme: Boolean = isSystemInDarkTheme(),
 // Dynamic color is available on Android 12+
 dynamicColor: Boolean = true,
 content: @Composable () -> Unit
) {
 val colorScheme = when {
 dynamicColor && Build.VERSION.SDK_INT >= Build.VERSION_CODES.S -> {
 val context = LocalContext.current
 if (darkTheme) dynamicDarkColorScheme(context) else
 dynamicLightColorScheme(context)
 }
 darkTheme -> DarkColorScheme
 else -> LightColorScheme
 }

 MaterialTheme(
 colorScheme = colorScheme,
 typography = Typography,
 content = content
)
}

```

### [app/src/main/java/com/example/antamvieclam/ui/theme/Type.kt](#)

```
package com.example.antamvieclam.ui.theme
```

```

import androidx.compose.material3.Typography
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.sp

// Set of Material typography styles to start with
val Typography = Typography(
 bodyLarge = TextStyle(
 fontFamily = FontFamily.Default,
 fontWeight = FontWeight.Normal,
 fontSize = 16.sp,
 lineHeight = 24.sp,
 letterSpacing = 0.5.sp
)
)

```

```

/* Other default text styles to override
titleLarge = TextStyle(
 fontFamily = FontFamily.Default,
 fontWeight = FontWeight.Normal,
 fontSize = 22.sp,
 lineHeight = 28.sp,
 letterSpacing = 0.sp
),
labelSmall = TextStyle(
 fontFamily = FontFamily.Default,
 fontWeight = FontWeight.Medium,
 fontSize = 11.sp,
 lineHeight = 16.sp,
 letterSpacing = 0.5.sp
)
*/
)

```

### [app/src/main/res/drawable/ic\\_launcher\\_background.xml](#)

```

<?xml version="1.0" encoding="utf-8"?>
<vector xmlns:android="http://schemas.android.com/apk/res/android"
 android:width="108dp"
 android:height="108dp"
 android:viewportWidth="108"
 android:viewportHeight="108">
 <path
 android:fillColor="#3DDC84"
 android:pathData="M0,0h108v108h-108z" />
 <path
 android:fillColor="#00000000"
 android:pathData="M9,0L9,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
 <path
 android:fillColor="#00000000"
 android:pathData="M19,0L19,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
 <path
 android:fillColor="#00000000"
 android:pathData="M29,0L29,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />

```

```
<path
 android:fillColor="#00000000"
 android:pathData="M39,0L39,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M49,0L49,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M59,0L59,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M69,0L69,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M79,0L79,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M89,0L89,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M99,0L99,108"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,9L108,9"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,19L108,19"
```

```
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,29L108,29"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,39L108,39"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,49L108,49"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,59L108,59"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,69L108,69"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,79L108,79"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,89L108,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M0,99L108,99"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
```

```
 android:fillColor="#00000000"
 android:pathData="M19,29L89,29"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M19,39L89,39"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M19,49L89,49"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M19,59L89,59"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M19,69L89,69"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M19,79L89,79"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M29,19L29,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M39,19L39,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M49,19L49,89"
 android:strokeWidth="0.8"
```

```
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M59,19L59,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M69,19L69,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
<path
 android:fillColor="#00000000"
 android:pathData="M79,19L79,89"
 android:strokeWidth="0.8"
 android:strokeColor="#33FFFFFF" />
</vector>
```

### [app/src/main/res/drawable/ic\\_launcher\\_foreground.xml](#)

```
<vector xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:aapt="http://schemas.android.com/aapt"
 android:width="108dp"
 android:height="108dp"
 android:viewportWidth="108"
 android:viewportHeight="108">
 <path android:pathData="M31,63.928c0,0 6.4,-11 12.1,-13.1c7.2,-2.6 26,-1.4 26,-1.4l38.1,38.1L107,108.928l-32,-1L31,63.928z">
 <aapt:attr name="android:fillColor">
 <gradient
 android:endX="85.84757"
 android:endY="92.4963"
 android:startX="42.9492"
 android:startY="49.59793"
 android:type="linear">
 <item
 android:color="#44000000"
 android:offset="0.0" />
 <item
 android:color="#00000000"
 android:offset="1.0" />
 </gradient>
 </aapt:attr>
```

```
</path>
<path
 android:fillColor="#FFFFFF"
 android:fillType="nonZero"
 android:pathData="M65.3,45.828l3.8,-6.6c0.2,-0.4 0.1,-0.9 -0.3,-1.1c-0.4,-0.2 -0.9,-0.1 -
1.1,0.3l-3.9,6.7c-6.3,-2.8 -13.4,-2.8 -19.7,0l-3.9,-6.7c-0.2,-0.4 -0.7,-0.5 -1.1,-0.3C38.8,38.328
38.7,38.828 38.9,39.228l3.8,6.6C36.2,49.428 31.7,56.028 31.63.928h46C76.3,56.028
71.8,49.428 65.3,45.828zM43.4,57.328c-0.8,0 -1.5,-0.5 -1.8,-1.2c-0.3,-0.7 -0.1,-1.5 0.4,-
2.1c0.5,-0.5 1.4,-0.7 2.1,-0.4c0.7,0.3 1.2,1 1.2,1.8C45.3,56.528 44.5,57.328
43.4,57.328L43.4,57.328zM64.6,57.328c-0.8,0 -1.5,-0.5 -1.8,-1.2s-0.1,-1.5 0.4,-2.1c0.5,-0.5
1.4,-0.7 2.1,-0.4c0.7,0.3 1.2,1 1.2,1.8C66.5,56.528 65.6,57.328 64.6,57.328L64.6,57.328z"
 android:strokeWidth="1"
 android:strokeColor="#00000000" />
</vector>
```

#### [app/src/main/res/mipmap-anydpi-v26/ic\\_launcher.xml](#)

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
 <background android:drawable="@drawable/ic_launcher_background" />
 <foreground android:drawable="@drawable/ic_launcher_foreground" />
 <monochrome android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

#### [app/src/main/res/mipmap-anydpi-v26/ic\\_launcher\\_round.xml](#)

```
<?xml version="1.0" encoding="utf-8"?>
<adaptive-icon xmlns:android="http://schemas.android.com/apk/res/android">
 <background android:drawable="@drawable/ic_launcher_background" />
 <foreground android:drawable="@drawable/ic_launcher_foreground" />
 <monochrome android:drawable="@drawable/ic_launcher_foreground" />
</adaptive-icon>
```

#### [app/src/main/res/values/colors.xml](#)

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
 <color name="purple_200">#FFBB86FC</color>
 <color name="purple_500">#FF6200EE</color>
 <color name="purple_700">#FF3700B3</color>
 <color name="teal_200">#FF03DAC5</color>
 <color name="teal_700">#FF018786</color>
 <color name="black">#FF000000</color>
 <color name="white">#FFFFFFFF</color>
</resources>
```

### **app/src/main/res/values/strings.xml**

```
<resources>
 <string name="app_name">AnTamViecLam</string>
</resources>
```

### **app/src/main/res/values/themes.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

 <style name="Theme.AnTamViecLam"
 parent="android:Theme.Material.Light.NoActionBar" />
</resources>
```

### **app/src/main/res/xml/backup\_rules.xml**

```
<?xml version="1.0" encoding="utf-8"?><!--
 Sample backup rules file; uncomment and customize as necessary.
 See https://developer.android.com/guide/topics/data/autobackup
 for details.

 Note: This file is ignored for devices older than API 31
 See https://developer.android.com/about/versions/12/backup-restore
-->
<full-backup-content>
 <!--
 <include domain="sharedpref" path="."/>
 <exclude domain="sharedpref" path="device.xml"/>
 -->
</full-backup-content>
```

### **app/src/main/res/xml/data\_extraction\_rules.xml**

```
<?xml version="1.0" encoding="utf-8"?><!--
 Sample data extraction rules file; uncomment and customize as necessary.
 See https://developer.android.com/about/versions/12/backup-restore#xml-changes
 for details.

-->
<data-extraction-rules>
 <cloud-backup>
 <!-- TODO: Use <include> and <exclude> to control what is backed up.
 <include .../>
 <exclude .../>
 -->
 </cloud-backup>
 <!--
 <device-transfer>
```

```
<include .../>
<exclude .../>
</device-transfer>
-->
</data-extraction-rules>

app/src/test/java/com/example/antamvieclam/ExampleUnitTest.kt
package com.example.antamvieclam

import org.junit.Test

import org.junit.Assert.*

/**
 * Example local unit test, which will execute on the development machine (host).
 *
 * See [testing documentation](http://d.android.com/tools/testing).
 */
class ExampleUnitTest {
 @Test
 fun addition_isCorrect() {
 assertEquals(4, 2 + 2)
 }
}
```

### [gradle/libs.versions.toml](#)

```
gradle/libs.versions.toml

[versions]
Plugins
crashlyticsPlugin = "3.0.1"
androidGradlePlugin = "8.4.1"
kotlin = "2.0.0"
ksp = "2.0.0-1.0.21"
googleServices = "4.4.2"
mapsSecretsPlugin = "2.0.1"
materialIconsExtended = "1.6.8"
trackasia = "2.0.2"
accompanist = "0.32.0"
secrets-gradle-plugin = "2.0.1"
cloudinary = "2.4.0"

SDKs
compileSdk = "34"
```

```
minSdk = "24"
targetSdk = "34"

THÊM CÁC THƯ VIỆN TRACKASIA
trackasia-sdk = { group = "com.track-asia", name = "trackasia-android-sdk", version.ref =
"trackasia" }
trackasia-annotation-plugin = { group = "com.track-asia", name = "trackasia-android-plugin-
annotation-v9", version.ref = "trackasia" }

Libraries
coreKtx = "1.13.1"
activityCompose = "1.9.0"
composeBom = "2024.06.00"
composeCompiler = "1.5.14"
hilt = "2.51.1"
lifecycle = "2.8.2"
room = "2.6.1"
firebaseBom = "33.1.1"
playServicesAuth = "21.2.0"
playServicesCoroutines = "1.8.1"
coroutines = "1.8.0"
retrofit = "2.9.0"
okhttp = "4.12.0"
navigation = "2.7.7"
hiltNavigation = "1.2.0"
coil = "2.6.0"
mapsCompose = "4.3.3" # <-- THÊM MỚI: Thư viện Maps cho Compose
playServicesMaps = "18.2.0" # <-- THÊM MỚI: Thư viện Google Maps SDK
playServicesLocation = "21.3.0" # <-- THÊM MỚI: Thư viện Location Services

Testing
junit = "4.13.2"
androidxJunit = "1.1.5"
espressoCore = "3.5.1"
firebaseAuth = "23.0.0"
viewbinding = "7.4.2"
transportBackendCct = "3.1.9"
transportApi = "3.0.0"

[libraries]
AndroidX Core
androidx-core-ktx = { group = "androidx.core", name = "core-ktx", version.ref = "coreKtx" }
```

```
androidx-lifecycle-runtime-ktx = { group = "androidx.lifecycle", name = "lifecycle-runtime-ktx", version.ref = "lifecycle" }
androidx-activity-compose = { group = "androidx.activity", name = "activity-compose", version.ref = "activityCompose" }

accompanist-navigation-animation = { group = "com.google.accompanist", name = "accompanist-navigation-animation", version.ref = "accompanist" }

Jetpack Compose
androidx-compose-bom = { group = "androidx.compose", name = "compose-bom", version.ref = "composeBom" }
androidx-compose-ui = { group = "androidx.compose.ui", name = "ui" }
androidx-compose-ui-graphics = { group = "androidx.compose.ui", name = "ui-graphics" }
androidx-compose-ui-tooling = { group = "androidx.compose.ui", name = "ui-tooling" }
androidx-compose-ui-tooling-preview = { group = "androidx.compose.ui", name = "ui-tooling-preview" }
androidx-compose-material3 = { group = "androidx.compose.material3", name = "material3" }
androidx-lifecycle-viewmodel-compose = { group = "androidx.lifecycle", name = "lifecycle-viewmodel-compose", version.ref = "lifecycle" }
androidx-lifecycle-runtime-compose = { group = "androidx.lifecycle", name = "lifecycle-runtime-compose", version.ref = "lifecycle" }
androidx-compose-material-icons-extended = { group = "androidx.compose.material", name = "material-icons-extended", version.ref = "materialIconsExtended" }

Image Loading - Coil
coil-compose = { module = "io.coil-kt:coil-compose", version.ref = "coil" }

Navigation
androidx-navigation-compose = { group = "androidx.navigation", name = "navigation-compose", version.ref = "navigation" }
androidx-hilt-navigation-compose = { group = "androidx.hilt", name = "hilt-navigation-compose", version.ref = "hiltNavigation" }

Dependency Injection - Hilt
hilt-android = { group = "com.google.dagger", name = "hilt-android", version.ref = "hilt" }
hilt-compiler = { group = "com.google.dagger", name = "hilt-compiler", version.ref = "hilt" }

Local Database - Room
androidx-room-runtime = { group = "androidx.room", name = "room-runtime", version.ref = "room" }
androidx-room-ktx = { group = "androidx.room", name = "room-ktx", version.ref = "room" }
```

```
androidx-room-compiler = { group = "androidx.room", name = "room-compiler", version.ref = "room" }

Remote - Firebase
cloudinary-android = { group = "com.cloudinary", name = "cloudinary-android", version.ref = "cloudinary" }
firebase-bom = { group = "com.google.firebaseio", name = "firebase-bom", version.ref = "firebaseBom" }
firebase-auth-ktx = { group = "com.google.firebaseio", name = "firebase-auth-ktx" }
firebase-firebase-ktx = { group = "com.google.firebaseio", name = "firebase-firebase-ktx" }
firebase-storage-ktx = { group = "com.google.firebaseio", name = "firebase-storage-ktx" }
firebase-messaging-ktx = { group = "com.google.firebaseio", name = "firebase-messaging-ktx" }
firebase-crashlytics-ktx = { group = "com.google.firebaseio", name = "firebase-crashlytics-ktx" }
play-services-auth = { group = "com.google.android.gms", name = "play-services-auth", version.ref = "playServicesAuth" }

THÊM MÓI: Google Maps & Location
maps-compose = { group = "com.google.maps.android", name = "maps-compose", version.ref = "mapsCompose" }
play-services-maps = { group = "com.google.android.gms", name = "play-services-maps", version.ref = "playServicesMaps" }
play-services-location = { group = "com.google.android.gms", name = "play-services-location", version.ref = "playServicesLocation" }
trackasia-sdk = { group = "io.github.track-asia", name = "android-sdk", version.ref = "trackasia" }
trackasia-annotation-plugin = { group = "io.github.track-asia", name = "android-plugin-annotation-v9", version = "2.0.1" }

Asynchronous - Coroutines
kotlinx-coroutines-core = { group = "org.jetbrains.kotlinx", name = "kotlinx-coroutines-core", version.ref = "coroutines" }
kotlinx-coroutines-android = { group = "org.jetbrains.kotlinx", name = "kotlinx-coroutines-android", version.ref = "coroutines" }
kotlinx-coroutines-play-services = { group = "org.jetbrains.kotlinx", name = "kotlinx-coroutines-play-services", version.ref = "playServicesCoroutines" }

Networking - Retrofit & OkHttp
retrofit = { group = "com.squareup.retrofit2", name = "retrofit", version.ref = "retrofit" }
converter-gson = { group = "com.squareup.retrofit2", name = "converter-gson", version.ref = "retrofit" }
logging-interceptor = { group = "com.squareup.okhttp3", name = "logging-interceptor",
```

```
version.ref = "okhttp" }

Testing
junit = { group = "junit", name = "junit", version.ref = "junit" }
androidx-junit = { group = "androidx.test.ext", name = "junit", version.ref = "androidxJunit" }
androidx-espresso-core = { group = "androidx.test.espresso", name = "espresso-core",
version.ref = "espressoCore" }
androidx-compose-ui-test-junit4 = { group = "androidx.compose.ui", name = "ui-test-junit4"
}
androidx-compose-ui-test-manifest = { group = "androidx.compose.ui", name = "ui-test-
manifest" }
firebase-auth = { group = "com.google.firebaseio", name = "firebase-auth", version.ref =
"firebaseAuth" }
androidx-viewbinding = { group = "androidx.databinding", name = "viewbinding",
version.ref = "viewbinding" }
transport-backend-cct = { group = "com.google.android.datatransport", name = "transport-
backend-cct", version.ref = "transportBackendCct" }
transport-api = { group = "com.google.android.datatransport", name = "transport-api",
version.ref = "transportApi" }
```

```
[plugins]
Khai báo các plugin của dự án
android-secrets-gradle-plugin = { id = "com.google.android.libraries.mapsplatform.secrets-
gradle-plugin", version.ref = "secrets-gradle-plugin" }
android-application = { id = "com.android.application", version.ref = "AndroidGradlePlugin"
}
android-library = { id = "com.android.library", version.ref = "AndroidGradlePlugin" }
kotlin-android = { id = "org.jetbrains.kotlin.android", version.ref = "kotlin" }
ksp = { id = "com.google.devtools.ksp", version.ref = "ksp" }
hilt = { id = "com.google.dagger.hilt.android", version.ref = "hilt" }
google-services = { id = "com.google.gms.google-services", version.ref = "googleServices" }
kotlin-compose-compiler = { id = "org.jetbrains.kotlin.plugin.compose", version.ref =
"kotlin" }
firebase-crashlytics = { id = "com.google.firebaseio.crashlytics", version.ref =
"crashlyticsPlugin" }
maps-secrets = { id = "com.google.android.libraries.mapsplatform.secrets-gradle-plugin",
version.ref = "mapsSecretsPlugin" } # <--- THÊM MỚI
```

```
[bundles]
Nhóm các thư viện thường đi chung với nhau để gọi cho gọn
compose = ["androidx-compose-ui", "androidx-compose-ui-graphics", "androidx-compose-
ui-tooling-preview", "androidx-compose-material3"]
```

```
room = ["androidx-room-runtime", "androidx-room-ktx"]
coroutines = ["kotlinx-coroutines-core", "kotlinx-coroutines-android"]
```

### **gradle/wrapper/gradle-wrapper.properties**

```
#Sun Nov 09 13:27:50 ICT 2025
distributionBase=GRADLE_USER_HOME
distributionPath=wrapper/dists
distributionUrl=https\://services.gradle.org/distributions/gradle-8.13-bin.zip
networkTimeout=10000
validateDistributionUrl=true
zipStoreBase=GRADLE_USER_HOME
zipStorePath=wrapper/dists
```