BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI

BÀI TẬP LỚN

PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG

ĐỀ TÀI: ỨNG DỤNG DỰ BÁO THỜI TIẾT

Giáo viên hướng dẫn: ThS. Kiều Tuấn Dũng

Sinh viên thực hiện:

STT	Mã sinh viên	Họ và tên	Lớp
1	2251061789	Lê Đức Hùng	64CNTT1
2	2251061788	Đỗ Trần Hùng	64CNTT1
3	2251061748	Thái Duy Đức	64CNTT1

Hà Nội, năm 2025

BỘ NÔNG NGHIỆP VÀ MÔI TRƯỜNG
TRƯỜNG ĐẠI HỌC THỦY LỢI

BÀI TẬP LỚN

PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG

ĐỀ TÀI: ỨNG DỤNG DỰ BÁO THỜI TIẾT

STT	Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
				Bằng Số	Bằng Chữ
1	2251061789	Lê Đức Hùng	12/05/2004		
2	2251061788	Đỗ Trần Hùng	09/12/2004		
3	2251061748	Thái Duy Đức	12/04/2004		

CÁN BỘ CHẤM THI

Hà Nội, năm 2025

LỜI NÓI ĐẦU

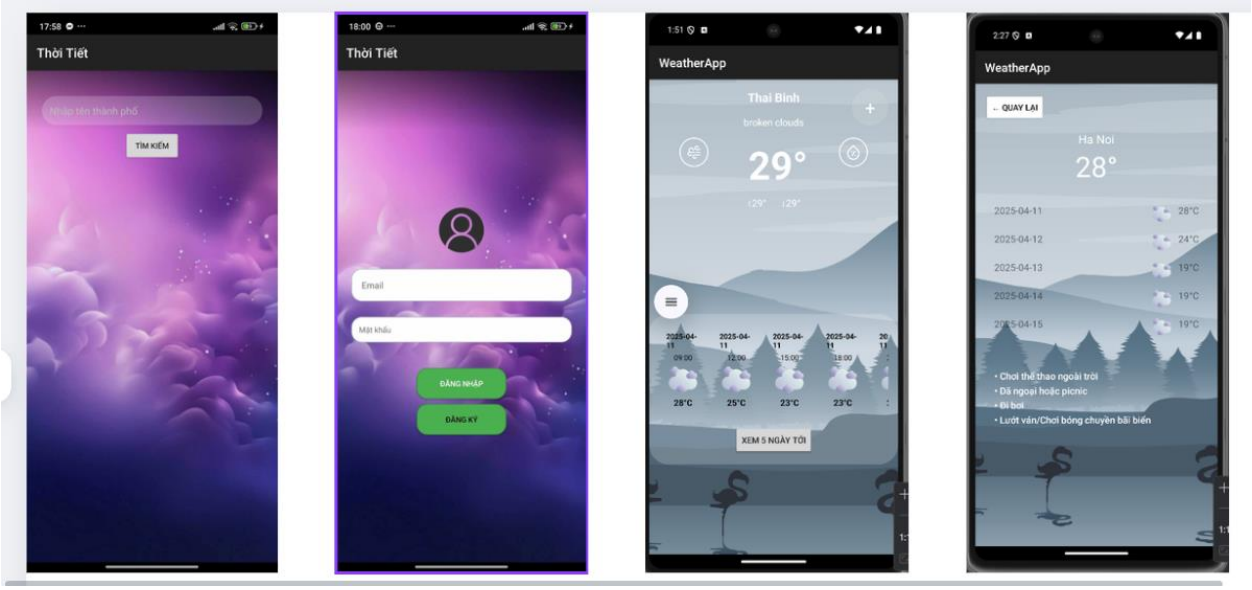
Trong thời đại công nghệ số phát triển mạnh mẽ như hiện nay, các ứng dụng di động đóng vai trò quan trọng trong việc hỗ trợ và nâng cao chất lượng cuộc sống con người. Trong đó, các ứng dụng dự báo thời tiết không chỉ giúp người dùng nắm bắt tình hình khí hậu nhanh chóng, mà còn góp phần hỗ trợ việc lên kế hoạch cho học tập, làm việc và các hoạt động ngoài trời một cách hiệu quả.

Với sự phổ biến rộng rãi của nền tảng Android và nhu cầu tra cứu thời tiết ngày càng tăng, việc xây dựng một ứng dụng dự báo thời tiết là hoàn toàn cần thiết và có tính ứng dụng cao. Đề tài "Dự báo thời tiết" được thực hiện nhằm mục tiêu tạo ra một ứng dụng Android thân thiện với người dùng, cung cấp thông tin thời tiết theo thời gian thực tại vị trí hiện tại hoặc các địa điểm mà người dùng quan tâm. Ứng dụng sử dụng dữ liệu từ các dịch vụ thời tiết công khai và tích hợp các công nghệ hiện đại như: API, kiến trúc MVVM, LiveData, Room, Retrofit,…

Thông qua quá trình nghiên cứu và xây dựng đề tài, nhóm thực hiện mong muốn không chỉ củng cố kiến thức về lập trình Android mà còn có cơ hội tiếp cận các công nghệ mới, rèn luyện tư duy hệ thống và kỹ năng phát triển phần mềm thực tiễn.

MỤC LỤC

DANH MỤC HÌNH ẢNH



DANH MỤC BẢNG BIỂU

STT	Chức năng	Mô tả	Giao diện / Tương tác
1	Tìm kiếm thành phố	Cho phép người dùng nhập tên thành phố và tìm kiếm thời tiết của thành phố đó.	Input Text + Button
2	Hiển thị thông tin thời tiết	Hiển thị thông tin thời tiết hiện tại, mô tả, nhiệt độ, độ ẩm, và hướng gió.	TextViews, Icons
3	Dự báo thời tiết theo giờ	Hiển thị dự báo thời tiết trong vòng 24 giờ tiếp theo với thông tin chi tiết theo từng giờ.	RecyclerView, CardViews
4	Dự báo 5 ngày tiếp theo	Hiển thị dự báo thời tiết trong 5 ngày tiếp theo, người dùng có thể nhấn nút để xem.	Button + RecyclerView
5	Thay đổi nền theo thời tiết	Thay đổi hình nền của ứng dụng tùy theo loại thời tiết (nắng, mưa, mây...).	ImageView (Background)

STT	API	Mô tả	Phương thức	Dữ liệu trả về
1	`getWeatherByCity`	Lấy thông tin thời tiết hiện tại của thành phố.	GET	`WeatherResponse`
2	`getHourlyForecast`	Lấy dự báo thời tiết theo giờ trong vòng 24 giờ.	GET	`ForecastResponse`
3	`getFiveDayForecast`	Lấy dự báo thời tiết cho 5 ngày tới.	GET	`ForecastResponse`

Tên lớp	Đối tượng	Mô tả
`WeatherResponse`	`city_name`, `temperature`, `description`	Đối tượng chứa thông tin thời tiết hiện tại của thành phố
`ForecastResponse`	`list`	Đối tượng chứa danh sách dự báo thời tiết theo giờ
`ForecastItem`	`dt_txt`, `main`, `weather`	Đối tượng mô tả một mục trong dự báo theo giờ
`DailyForecast`	`date`, `temp`, `icon`	Đối tượng mô tả dự báo thời tiết cho 1 ngày

DANH MỤC CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	RWD	Responsive Web Design
2	MVVM	Model – View – ViewModel
3	API	Application Programming Interface
4	GPS	Global Positioning System

Chương 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu về đề tài

Trong thời đại công nghệ số hiện nay, nhu cầu nắm bắt thông tin thời tiết một cách nhanh chóng, chính xác và tiện lợi ngày càng trở nên quan trọng đối với đời sống hằng ngày cũng như trong các lĩnh vực sản xuất, nông nghiệp, du lịch và giao thông. Nhận thức được điều đó, đề tài "Dự báo thời tiết" được phát triển dưới dạng ứng dụng Android, sử dụng ngôn ngữ Java/Kotlin kết hợp với các thư viện và công nghệ hiện đại nhằm cung cấp cho người dùng một ứng dụng tiện ích, hỗ trợ theo dõi tình hình thời tiết theo thời gian thực, dự báo trong nhiều ngày tới, kèm theo các thông tin chi tiết như nhiệt độ, độ ẩm, lượng mưa, tốc độ gió và chất lượng không khí.

Ứng dụng hướng tới giao diện thân thiện, dễ sử dụng, tích hợp các công nghệ hiện đại như API thời tiết, định vị GPS và trí tuệ nhân tạo để cải thiện độ chính xác trong dự báo. Không chỉ đơn thuần cung cấp thông tin, ứng dụng còn có khả năng đưa ra cảnh báo thời tiết cực đoan, giúp người dùng chủ động hơn trong việc lên kế hoạch và bảo vệ sức khỏe.

Với mong muốn mang lại một công cụ hữu ích, tiện lợi và có giá trị thực tiễn cao, đề tài "Dự báo thời tiết" không chỉ là một sản phẩm công nghệ mà còn là sự kết hợp giữa kỹ thuật lập trình và nhu cầu thiết thực của xã hội hiện đại.

1.2. Mục tiêu của đề tài

Mục tiêu chính của đề tài "Dự báo thời tiết" là xây dựng một ứng dụng di động trên nền tảng Android, sử dụng Java hoặc Kotlin, có khả năng thu thập và hiển thị thông tin thời tiết một cách nhanh chóng, chính xác, trực quan và thân thiện với người dùng. Cụ thể, đề tài hướng đến các mục tiêu kỹ thuật sau:

1. Thiết kế và xây dựng giao diện người dùng (UI/UX) theo tiêu chuẩn Material Design, đảm bảo tính thẩm mỹ và dễ sử dụng.
2. Tích hợp API thời tiết (như OpenWeatherMap API) để lấy dữ liệu thời tiết hiện tại và dự báo trong tương lai (theo giờ, theo ngày).
3. Xử lý dữ liệu thời tiết từ API dưới dạng JSON, chuyển đổi sang mô hình dữ liệu phù hợp trong Java/Kotlin, và hiển thị bằng các thành phần giao diện như RecyclerView, CardView, ViewPager, biểu đồ,…
4. Sử dụng định vị GPS để tự động xác định vị trí hiện tại của người dùng và hiển thị thông tin thời tiết tương ứng.
5. Áp dụng kiến trúc MVVM để phân tách logic nghiệp vụ, giao diện và dữ liệu, giúp ứng dụng dễ mở rộng và bảo trì.

6. Lưu trữ dữ liệu cục bộ (như lịch sử địa điểm, địa điểm yêu thích) sử dụng Room Database.
7. Hỗ trợ cảnh báo thời tiết xấu hoặc các điều kiện đặc biệt (nhiệt độ cao, mưa lớn, gió mạnh,...).
8. Tối ưu hiệu năng ứng dụng, bao gồm: tải dữ liệu nhanh, giảm thiểu tiêu thụ pin và băng thông, quản lý bộ nhớ hiệu quả.
9. Tích hợp các thư viện và công cụ hiện đại như Retrofit, Glide/Picasso, LiveData, ViewModel,… trong quá trình phát triển.

1.3. Phạm vi của đề tài

Đề tài "Dự báo thời tiết" tập trung vào việc thiết kế và phát triển một ứng dụng di động hoạt động trên nền tảng Android, sử dụng ngôn ngữ Java hoặc Kotlin, nhằm cung cấp cho người dùng thông tin thời tiết một cách chính xác và trực quan. Phạm vi của đề tài bao gồm các nội dung chính sau:

1. Nền tảng triển khai:
    - Hệ điều hành Android (API từ 21 trở lên).
    - Thiết bị hỗ trợ: điện thoại và máy tính bảng Android.
    - Ngôn ngữ lập trình: Java hoặc Kotlin.
2. Tính năng chính:
    - Hiển thị thông tin thời tiết hiện tại tại vị trí người dùng.
    - Cho phép tìm kiếm thời tiết tại các thành phố hoặc khu vực khác.
    - Dự báo thời tiết theo giờ, theo ngày (3–7 ngày).
    - Cung cấp các chỉ số như: nhiệt độ, độ ẩm, tốc độ gió, áp suất, chỉ số UV,…
    - Hiển thị biểu tượng thời tiết và hình ảnh minh họa tương ứng với trạng thái thời tiết.
    - Giao diện thân thiện, hỗ trợ chế độ sáng/tối (dark mode).
3. Phạm vi kỹ thuật:
    - Sử dụng API của dịch vụ bên thứ ba (ví dụ: OpenWeatherMap).
    - Áp dụng kiến trúc MVVM.
    - Sử dụng thư viện Retrofit để gọi API và xử lý phản hồi.

- Sử dụng LiveData và ViewModel để quản lý dữ liệu và vòng đời.
- Sử dụng Room Database để lưu trữ cục bộ (nếu có).
- Định vị vị trí người dùng bằng FusedLocationProvider (Google Location Services).
- Tối ưu hóa hiệu năng: xử lý bất đồng bộ, cache dữ liệu, tiết kiệm pin.

4. Không bao gồm trong phạm vi đề tài:
    - Không xử lý phân tích dữ liệu dự báo chuyên sâu bằng thuật toán học máy.
    - Không xây dựng hệ thống backend riêng mà sử dụng dữ liệu từ dịch vụ thời tiết công khai.
    - Không hỗ trợ đa nền tảng (iOS hoặc web) trong phiên bản hiện tại.

1.4 Phân chia nhiệm vụ

<<Bảng phân chia nhiệm vụ>>

STT	Họ và tên	Nhiệm vụ
1	Lê Đức Hùng	Chịu trách nhiệm về logic xử lý dữ liệu và Firebase Authentication, xử lý sự kiện người dùng
2	Đỗ Trần Hùng	Đảm bảo giao diện app trực quan, đẹp mắt, và tương tác với API OpenWeatherMap và lý sự kiện người dùng
3	Thái Duy Đức	Đảm bảo chất lượng sản phẩm thông qua kiểm thử, quản lý dự án và tích hợp các tính năng nâng cao.

Chương 2. KIẾN TRÚC VÀ CÔNG NGHỆ

2.1. Kiến trúc hệ thống

- Ứng dụng "Dự báo thời tiết" được xây dựng theo mô hình client-server, trong đó:
  - ✓ Client là ứng dụng Android do người dùng sử dụng, được phát triển bằng ngôn ngữ Java/Kotlin.
  - ✓ Server là hệ thống cung cấp dữ liệu thời tiết, ví dụ như OpenWeatherMap, thông qua giao tiếp RESTful API.
- Hệ thống sử dụng kiến trúc MVVM (Model – View – ViewModel) để tách biệt ba phần chính:
  - ✓ Model: xử lý dữ liệu từ API và quản lý dữ liệu cục bộ (Room Database).
  - ✓ ViewModel: trung gian giữa View và Model, xử lý logic hiển thị và cung cấp dữ liệu theo vòng đời của UI.
  - ✓ View: giao diện người dùng, được cập nhật tự động thông qua LiveData.

2.2. Giới thiệu về Công nghệ phát triển

Các công nghệ và thư viện được sử dụng trong ứng dụng bao gồm:

- Ngôn ngữ lập trình:
  - ✓ Java hoặc Kotlin – hai ngôn ngữ chính được hỗ trợ trên Android.
- Giao tiếp với API:
  - ✓ Retrofit: thư viện mạnh mẽ dùng để gọi API RESTful và xử lý phản hồi JSON.
  - ✓ Gson/Moshi: dùng để chuyển đổi dữ liệu JSON thành đối tượng Java/Kotlin.
- Quản lý dữ liệu và vòng đời:
  - ✓ LiveData: giúp giao diện tự động cập nhật khi dữ liệu thay đổi.
  - ✓ ViewModel: quản lý dữ liệu và logic tương tác giữa UI và Model.
  - ✓ Room: cơ sở dữ liệu cục bộ để lưu trữ các địa điểm yêu thích, lịch sử.
- Xử lý ảnh:
  - ✓ Glide hoặc Picasso: dùng để tải và hiển thị hình ảnh biểu tượng thời tiết.
- Định vị và vị trí:

- ✓ FusedLocationProviderClient: lấy vị trí người dùng thông qua Google Play Services.

- Thiết kế giao diện:
  - ✓ Material Design Components: tuân thủ tiêu chuẩn thiết kế giao diện hiện đại của Google.
  - ✓ ConstraintLayout, RecyclerView, ViewPager,...
- Xử lý bất đồng bộ:
  - ✓ Coroutines (Kotlin) hoặc AsyncTask/Executors (Java).

Chương 3. XÂY DỰNG ỨNG DỤNG

3.1. Thiết kế Figma

<<Link Gitub và ảnh kết xuất Figma các màn hình>>

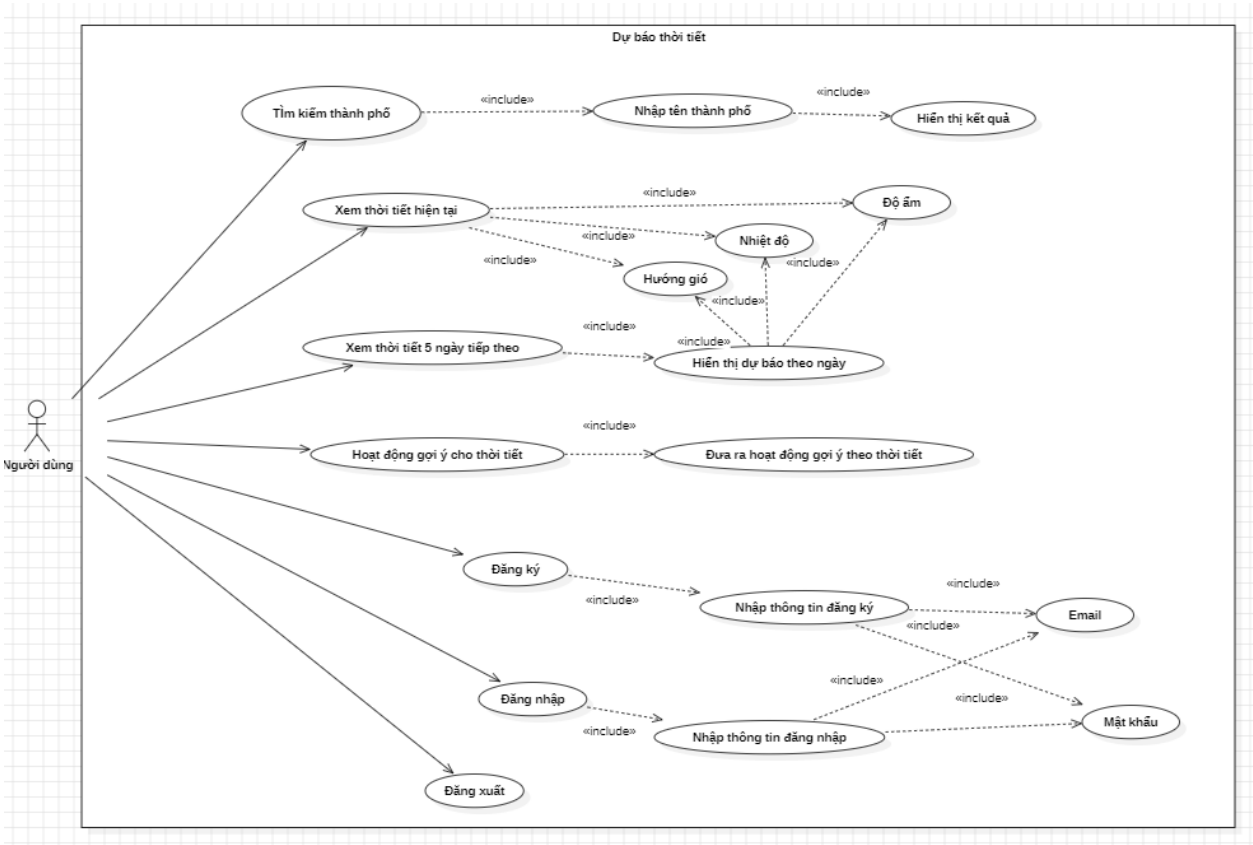

Thái Bình
25°
5 thg 4 Ngày mai
21° / 23°
6 thg 4 Chủ nhật
21° / 23°
7 thg 4 Thứ 2
21° / 23°
8 thg 4 Thứ 3
21° / 23°
9 thg 4 Thứ 4
21° / 23°
Một số hoạt động gợi ý


Thái Bình
Mostly Cloudy
25°
26°
24°
Fri 9am
Fri 0pm
Fri 3am
Fri 6am
Fri 9am
25°C
26°C
23°C
21°C
25°C
Dự báo 5 ngày tiếp theo

Link github: https://github.com/duchungvippro/B-i-t-p-Mobile-Dev

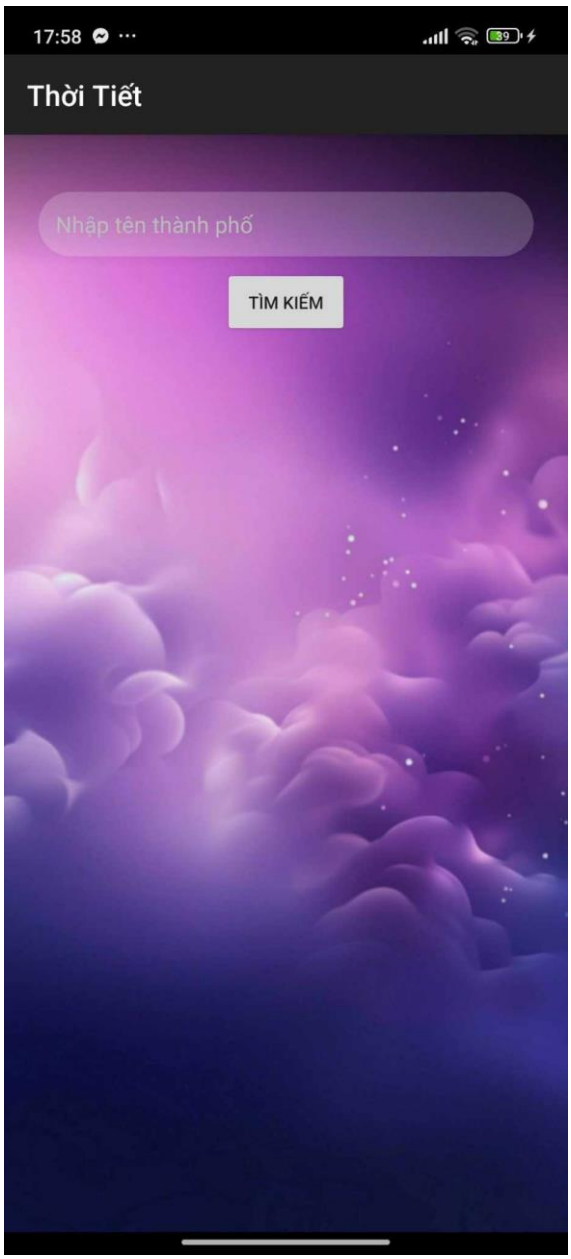
3.2. Thiết kế CSDL

<<Lược đồ>>

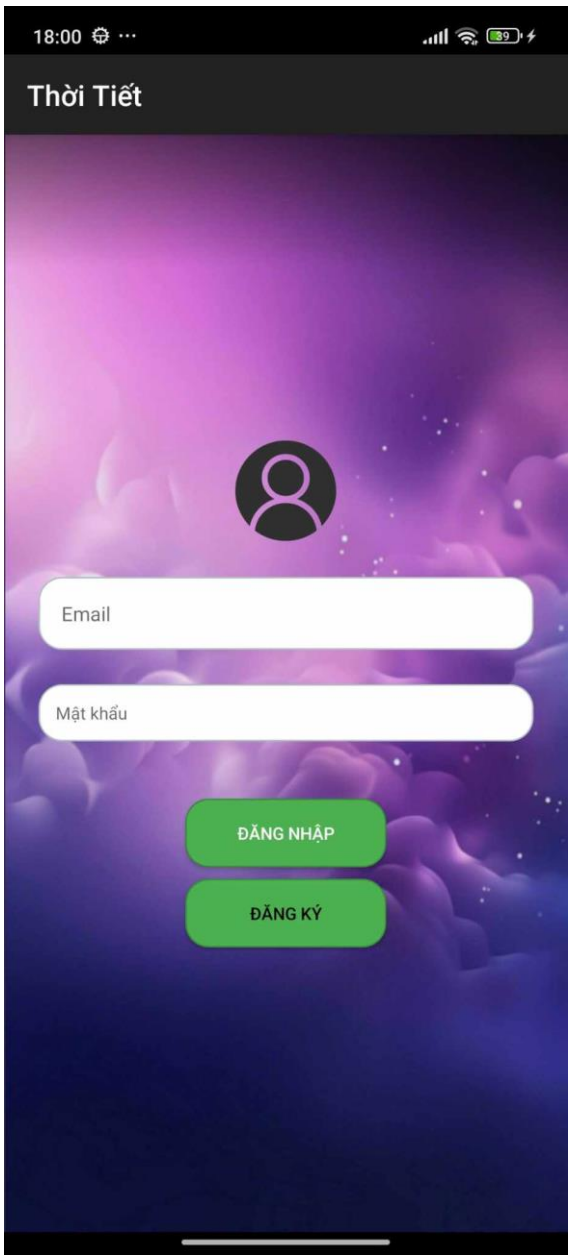


3.3. Giao diện ứng dụng

3.3.1. Màn hình



3.3.2. Đăng nhập, đăng ký



3.3.3. Dự báo theo giờ

3.3.4. Hoạt động gợi ý

3.4. Code minh họa các chức năng cốt lõi

3.4.1. MainActivity

```
package com.example.weatherapp


import android.content.Intent
import android.os.Bundle
import android.util.Log
import android.view.View
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import com.example.weatherapp.databinding.ActivityMainBinding
import retrofit2.*
import retrofit2.converter.gson.GsonConverterFactory
```

```
class MainActivity : AppCompatActivity() {
    private lateinit var binding: ActivityMainBinding
    private val apiKey = "d754023544590c6ced61ca8f5582dce3"

    // Khai báo biến toàn cục cho dự báo 5 ngày
    private val fiveDaysForecastData = mutableListOf<DailyForecast>()
    private var currentSearchedCity: String = ""

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        // Khởi tạo binding
        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        // Khởi tạo Retrofit và service
        val retrofit = Retrofit.Builder()
            .baseUrl("https://api.openweathermap.org/data/2.5/")
            .addConverterFactory(GsonConverterFactory.create())
            .build()

        val service = retrofit.create(WeatherService::class.java)

        // Set up RecyclerView layout managers
        binding.recyclerHourly.layoutManager = LinearLayoutManager(this,
LinearLayoutManager.HORIZONTAL, false)

        // Hiển thị giao diện tìm kiếm
        showSearchUI(true)

        // Cài đặt các OnClickListener
        binding.btnSearch.setOnClickListener {
            val city = binding.cityInput.text.toString().trim()
            if (city.isEmpty()) {
                Toast.makeText(this, getString(R.string.enter_city),
Toast.LENGTH_SHORT).show()
                return@setOnClickListener
            }

            val lang = resources.configuration.locales.get(0).language
            currentSearchedCity = city
```

```
            // Gọi API thời tiết
            service.getWeatherByCity(city, apiKey, lang = lang)
                .enqueue(object : Callback<WeatherResponse> {
                    override fun onResponse(
                        call: Call<WeatherResponse>,
                        response: Response<WeatherResponse>
                    ) {
                        if (response.isSuccessful && response.body() != null) {
                            val data = response.body()!!

                            // Cập nhật giao diện với dữ liệu nhận được
                            binding.tvCity.text = data.name
                            binding.tvDescription.text = data.weather[0].description
                            binding.tvTemp.text = "${data.main.temp.toInt()}°"
                            binding.tvTempMax.text = "↑${data.main.tempMax.toInt()}°"
                            binding.tvTempMin.text = "↓${data.main.tempMin.toInt()}°"

                            setBackgroundByWeather(data.weather[0].description)
                            showSearchUI(false)

                            // Dự báo theo giờ
                            service.getHourlyForecast(city, apiKey, lang = lang)
                                .enqueue(object : Callback<ForecastResponse> {
                                    override fun onResponse(
                                        call: Call<ForecastResponse>,
                                        response: Response<ForecastResponse>
                                    ) {
                                        if (response.isSuccessful) {
                                            val forecastList =
response.body()?.list?.take(5) ?: return
                                            val adapter = HourlyAdapter(forecastList)
                                            binding.recyclerHourly.adapter = adapter
                                        }
                                    }

                                    override fun onFailure(call: Call<ForecastResponse>,
t: Throwable) {
                                        Log.e("FORECAST", "Error loading hourly
forecast", t)
                                    }
                                })
```

```
                        // Gọi dự báo 5 ngày
                        binding.btnFiveDay.setOnClickListener {
                            if (fiveDaysForecastData.isNotEmpty()) {
                                val intent = Intent(this@MainActivity,
FiveDaysForecastActivity::class.java)
                                intent.putExtra("forecastList",
ArrayList(fiveDaysForecastData))
                                intent.putExtra("city_name", currentSearchedCity)
                                intent.putExtra("current_temp",
binding.tvTemp.text.toString())

                                startActivity(intent)
                            } else {
                                Toast.makeText(this@MainActivity,
getString(R.string.city_not_found), Toast.LENGTH_SHORT).show()
                            }
                        }

                        // Load dự báo 5 ngày
                        loadFiveDayForecast(city)
                    } else {
                        Toast.makeText(
                            this@MainActivity,
                            getString(R.string.city_not_found),
                            Toast.LENGTH_SHORT
                        ).show()
                    }
                }

                override fun onFailure(call: Call<WeatherResponse>, t: Throwable) {
                    Log.e("WEATHER_API_ERROR", "Failed API call", t)
                    Toast.makeText(
                        this@MainActivity,
                        getString(R.string.api_error),
                        Toast.LENGTH_SHORT
                    ).show()
                }
            })
        }

        binding.btnAddCity.setOnClickListener {
            showSearchUI(true)
```

```
            binding.cityInput.setText("")
        }
    }

    private fun showSearchUI(show: Boolean) {
        binding.cityInput.visibility = if (show) View.VISIBLE else View.GONE
        binding.btnSearch.visibility = if (show) View.VISIBLE else View.GONE

        binding.tvCity.visibility = if (show) View.GONE else View.VISIBLE
        binding.tvDescription.visibility = if (show) View.GONE else View.VISIBLE
        binding.tvTemp.visibility = if (show) View.GONE else View.VISIBLE
        binding.tvTempMax.visibility = if (show) View.GONE else View.VISIBLE
        binding.tvTempMin.visibility = if (show) View.GONE else View.VISIBLE

        binding.hourlyContainer.visibility = if (show) View.GONE else View.VISIBLE
        binding.btnAddCity.visibility = if (show) View.GONE else View.VISIBLE

        binding.iconWind.visibility = if (show) View.GONE else View.VISIBLE
        binding.iconHumidity.visibility = if (show) View.GONE else View.VISIBLE
    }

    private fun setBackgroundByWeather(description: String) {
        val desc = description.lowercase()

        val bgRes = when {
            "clear" in desc || "sun" in desc || "nắng" in desc -> R.drawable.bg_day
            "rain" in desc || "drizzle" in desc || "mưa" in desc -> R.drawable.bg_rain
            "cloud" in desc || "mây" in desc -> R.drawable.bg_clouds
            else -> R.drawable.bg_night
        }

        binding.bgImage.setImageResource(bgRes)
    }

    private fun loadFiveDayForecast(city: String) {
        val retrofit = Retrofit.Builder()
            .baseUrl("https://api.openweathermap.org/data/2.5/")
            .addConverterFactory(GsonConverterFactory.create())
            .build()

        val service = retrofit.create(WeatherService::class.java)
        val lang = resources.configuration.locales.get(0).language
```

```
        service.getFiveDayForecast(city, apiKey, lang = lang).enqueue(object :
Callback<ForecastResponse> {
            override fun onResponse(call: Call<ForecastResponse>, response:
Response<ForecastResponse>) {
                if (response.isSuccessful && response.body() != null) {
                    val allForecasts = response.body()!!.list

                    val dailyList = mutableListOf<DailyForecast>()
                    for (item in allForecasts) {
                        val date = item.dt_txt.substring(0, 10)
                        if (dailyList.none { it.date == date } && dailyList.size < 5) {
                            dailyList.add(
                                DailyForecast(
                                    date = date,
                                    temp = item.main.temp,
                                    icon = item.weather[0].icon
                                )
                            )
                        }
                    }

                    // Lưu dự báo vào `fiveDaysForecastData`
                    fiveDaysForecastData.clear()
                    fiveDaysForecastData.addAll(dailyList)

                    val adapter = DailyForecastAdapter(dailyList)
                }
            }

            override fun onFailure(call: Call<ForecastResponse>, t: Throwable) {
                Log.e("FIVEDAY", "Error loading 5 day forecast", t)
            }
        })
    }
}
```

3.4.2. DailyForecastAdapter

```
package com.example.weatherapp

import androidx.recyclerview.widget.RecyclerView
import android.view.LayoutInflater
```

```
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import com.bumptech.glide.Glide

class DailyForecastAdapter(private val list: List<DailyForecast>) :
    RecyclerView.Adapter<DailyForecastAdapter.DailyViewHolder>() {

    class DailyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val date = view.findViewById<TextView>(R.id.tvDate)
        val temp = view.findViewById<TextView>(R.id.tvTemp)
        val icon = view.findViewById<ImageView>(R.id.imgWeather)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): DailyViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.item_hourly, parent, false)
        return DailyViewHolder(view)
    }

    override fun onBindViewHolder(holder: DailyViewHolder, position: Int) {
        val item = list[position]
        holder.date.text = item.date
        holder.temp.text = "${item.temp.toInt()}°C"

        // Sử dụng Glide để tải icon từ OpenWeatherMap
        val iconRes = getIconResource(item.icon)
        holder.icon.setImageResource(iconRes)

    }

   // Thêm hàm này trong DailyForecastAdapter.kt
    private fun getIconResource(icon: String): Int {
        return when (icon) {
            "01d" -> R.drawable.ic_sun   // Icon trời nắng ban ngày
            "01n" -> R.drawable.ic_sun // Icon trời nắng ban đêm
            "02d" -> R.drawable.ic_few_clouds_day  // Icon ít mây ban ngày
            "02n" -> R.drawable.ic_cloud // Icon ít mây ban đêm
            "03d", "03n" -> R.drawable.ic_cloud // Icon mây rải rác
            "04d", "04n" -> R.drawable.ic_cloud // Icon mây phủ
            "09d", "09n" -> R.drawable.ic_rain  // Icon mưa
```

```
            "10d", "10n" -> R.drawable.ic_rain  // Icon mưa
            "11d", "11n" -> R.drawable.ic_storm // Icon bão
            "13d", "13n" -> R.drawable.ic_snow // Icon tuyết

            else -> R.drawable.ic_sun // Icon mặc định nếu không có match
        }
    }

    override fun getItemCount() = list.size
}
```

3.4.3. FiveDaysAdapter

```
package com.example.weatherapp

import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import android.widget.ImageView
import android.widget.TextView
import androidx.recyclerview.widget.RecyclerView

class FiveDaysAdapter(private val forecastList: List<DailyForecast>) :
    RecyclerView.Adapter<FiveDaysAdapter.ForecastViewHolder>() {

    inner class ForecastViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val dateText: TextView = view.findViewById(R.id.tvDate)
        val tempText: TextView = view.findViewById(R.id.tvTemp)
        val iconImage: ImageView = view.findViewById(R.id.imgWeather)
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ForecastViewHolder
{
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.item_forecast, parent, false)
        return ForecastViewHolder(view)
    }

    override fun onBindViewHolder(holder: ForecastViewHolder, position: Int) {
        val forecast = forecastList[position]
        holder.dateText.text = forecast.date
        holder.tempText.text = "${forecast.temp.toInt()}°C"
```

```
        val iconRes = getIconResource(forecast.icon)
        holder.iconImage.setImageResource(iconRes)
    }

    private fun getIconResource(icon: String): Int {
        return when (icon) {
            "01d" -> R.drawable.ic_sun
            "01n" -> R.drawable.ic_sun
            "02d" -> R.drawable.ic_few_clouds_day
            "02n" -> R.drawable.ic_cloud
            "03d", "03n" -> R.drawable.ic_cloud
            "04d", "04n" -> R.drawable.ic_cloud
            "09d", "09n" -> R.drawable.ic_rain
            "10d", "10n" -> R.drawable.ic_rain
            "11d", "11n" -> R.drawable.ic_storm
            "13d", "13n" -> R.drawable.ic_snow
            else -> R.drawable.ic_sun
        }
    }

    override fun getItemCount() = forecastList.size
}
```

3.4.4. FiveDaysForecastActivity

```
package com.example.weatherapp

import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import com.example.weatherapp.databinding.ActivityFiveDaysForecastBinding

class FiveDaysForecastActivity : AppCompatActivity() {

    private lateinit var binding: ActivityFiveDaysForecastBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityFiveDaysForecastBinding.inflate(layoutInflater)
        setContentView(binding.root)
```

```
        val forecastList = intent.getSerializableExtra("forecastList") as?
ArrayList<DailyForecast>
        val cityName = intent.getStringExtra("city_name")
        val currentTemp = intent.getStringExtra("current_temp")

        if (forecastList == null) {
            Toast.makeText(this, "Không có dữ liệu dự báo thời tiết",
Toast.LENGTH_SHORT).show()
            return
        }

        // Gán tên thành phố
        binding.tvLocation.text = cityName

        // Gán nhiệt độ hiện tại
        binding.tvCurrentTemp.text = currentTemp

        // Gợi ý hoạt động theo nhiệt độ
        val tempValue = currentTemp?.replace("°", "")?.toIntOrNull() ?: 0
        val activitySuggestions = suggestActivity(tempValue)
        val suggestionText = activitySuggestions.joinToString(separator = "\n") { "• $it"
}
        binding.tvSuggestion.text = suggestionText

        // Thiết lập danh sách 5 ngày
        binding.recyclerViewFiveDays.layoutManager = LinearLayoutManager(this)
        binding.recyclerViewFiveDays.adapter = FiveDaysAdapter(forecastList)

        // Nút quay lại
        binding.btnBack.setOnClickListener {
            finish()
        }
    }


   private fun suggestActivity(tempC: Int): List<String> {
        return when {
            tempC < 10 -> listOf(
                "Đọc sách trong nhà",
                "Uống trà nóng",
                "Xem phim",
```

```
                "Nghe nhạc thư giãn"
            )
            tempC in 10..20 -> listOf(
                "Đi dạo công viên",
                "Uống café với bạn bè",
                "Chụp ảnh ngoài trời",
                "Tản bộ cùng thú cưng"
            )
            tempC in 21..30 -> listOf(
                "Chơi thể thao ngoài trời",
                "Dã ngoại hoặc picnic",
                "Đi bơi",
                "Lướt ván/Chơi bóng chuyền bãi biển"
            )
            else -> listOf(
                "Ở trong phòng máy lạnh",
                "Uống nước mát và nghỉ ngơi",
                "Đi bơi hoặc công viên nước",
                "Tránh vận động mạnh ngoài trời"
            )
        }
    }
}
```

3.4.5. activity_five_days_forecast.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@drawable/bg_clouds"
    android:padding="16dp">

    <!-- Nút quay lại -->
    <Button
        android:id="@+id/btnBack"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Quay Lai"
        android:backgroundTint="@android:color/white"
        android:textColor="@android:color/black"
```

```
        android:layout_marginBottom="16dp"/>

    <!-- Địa điểm -->
    <TextView
        android:id="@+id/tvLocation"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Thái Bình"
        android:textSize="20sp"
        android:textColor="@android:color/white"
        android:layout_gravity="center"/>

    <!-- Nhiệt độ -->
    <TextView
        android:id="@+id/tvCurrentTemp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="25°"
        android:textSize="48sp"
        android:textColor="@android:color/white"
        android:layout_gravity="center"
        android:layout_marginBottom="16dp"/>

    <!-- Danh sách dự báo 5 ngày -->
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerViewFiveDays"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="8dp"
        android:layout_marginBottom="16dp"/>

    <!-- Gợi ý hoạt động -->
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:padding="16dp"
        android:orientation="vertical"
        android:gravity="start">

        <TextView
            android:id="@+id/tvSuggestion"
            android:layout_width="match_parent"
```

```
            android:layout_height="wrap_content"
            android:text="Một số hoạt động gợi ý"
            android:textSize="16sp"
            android:textColor="@android:color/white"
            android:lineSpacingExtra="6dp"
            android:fontFamily="sans-serif-medium" />
    </LinearLayout>



</LinearLayout>
```

3.4.6. activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/rootLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <!-- Ảnh nền theo mô tả thời tiết -->
    <ImageView
        android:id="@+id/bgImage"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scaleType="centerCrop"
        android:src="@drawable/bg_day"
        android:contentDescription="@string/app_name" />



    <!-- Nút thêm thành phố (dấu +) -->
    <Button
        android:id="@+id/btnAddCity"
        android:layout_width="60dp"
        android:layout_height="60dp"
        android:text="+"
        android:textSize="30sp"
        android:gravity="center"
        android:background="@drawable/bg_rounded"
        android:textColor="@android:color/white"
```

```
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_margin="16dp" />

    <!-- Biểu tượng gió -->
    <ImageView
        android:id="@+id/iconWind"
        android:layout_width="48dp"
        android:layout_height="48dp"
        android:src="@drawable/ic_wind"
        android:visibility="gone"
        app:layout_constraintTop_toTopOf="@id/resultContainer"
        app:layout_constraintStart_toStartOf="parent"
        android:layout_marginStart="50dp"
        android:layout_marginTop="80dp" />

    <!-- Biểu tượng độ ẩm -->
    <ImageView
        android:id="@+id/iconHumidity"
        android:layout_width="48dp"
        android:layout_height="48dp"
        android:src="@drawable/ic_humidity"
        android:visibility="gone"
        android:layout_marginTop="80dp"
        android:layout_marginEnd="50dp"
        app:layout_constraintTop_toTopOf="@id/resultContainer"
        app:layout_constraintEnd_toEndOf="parent" />

    <!-- Nhập thành phố -->
    <EditText
        android:id="@+id/cityInput"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="Nhập tên thành phố"
        android:padding="12dp"
        android:textColor="@android:color/white"
        android:textColorHint="#BDBDBD"
        android:background="@drawable/bg_rounded"
        android:layout_marginTop="40dp"
        android:layout_marginHorizontal="24dp"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent" />
```

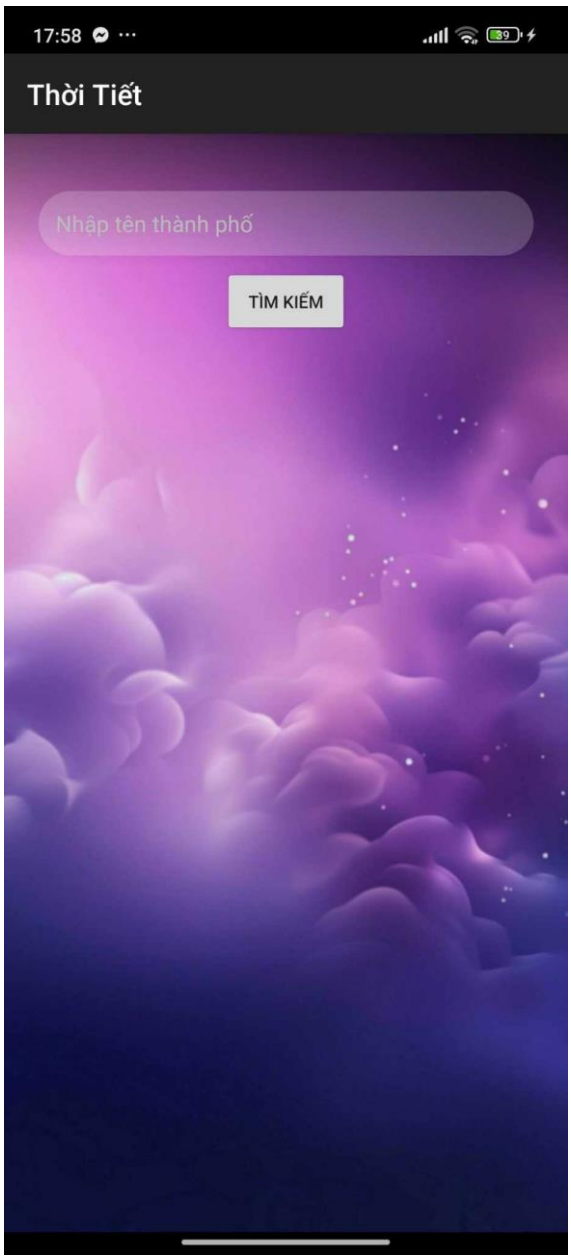
```
<!-- Nút tìm kiếm -->
<Button
    android:id="@+id/btnSearch"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="TÌM KIẾM"
    android:layout_marginTop="8dp"
    app:layout_constraintTop_toBottomOf="@id/cityInput"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

<!-- Khối kết quả -->
<LinearLayout
    android:id="@+id/resultContainer"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:gravity="center_horizontal"
    android:orientation="vertical"
    app:layout_constraintTop_toBottomOf="@id/btnSearch"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent">

    <!-- Tên thành phố -->
    <TextView
        android:id="@+id/tvCity"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="4dp"
        android:text="Hà Nội"
        android:textColor="@android:color/white"
        android:textSize="20sp"
        android:textStyle="bold" />

    <!-- Mô tả thời tiết -->
    <TextView
        android:id="@+id/tvDescription"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="12dp"
        android:layout_marginBottom="8dp"
        android:gravity="center"
```

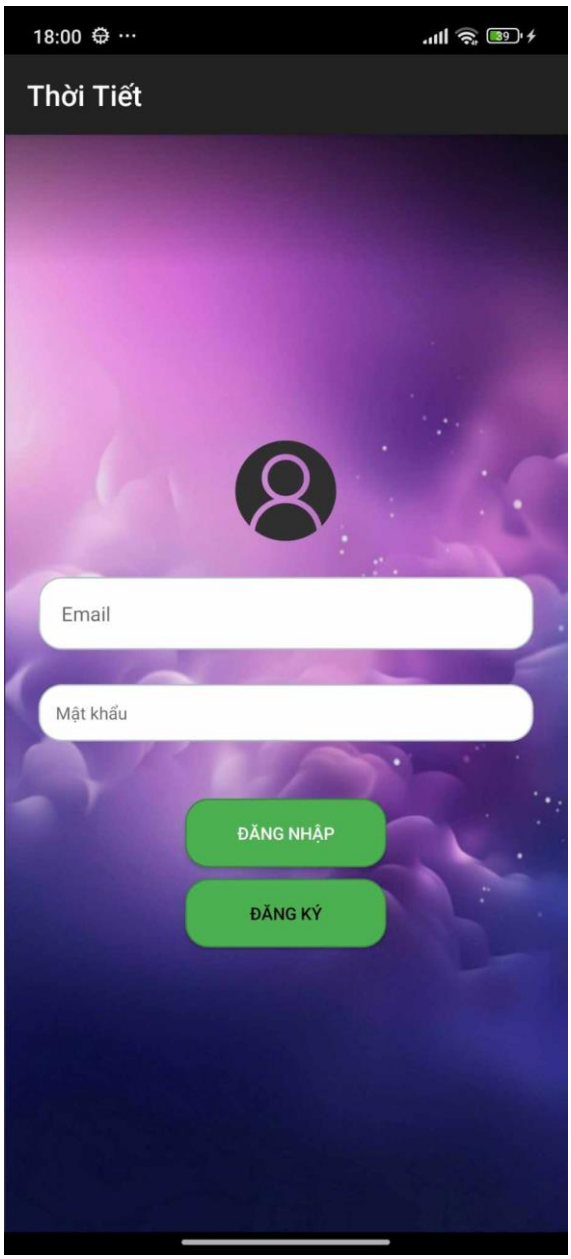
```
            android:text="Mostly Cloudy"
            android:textColor="@android:color/white"
            android:textSize="16sp" />

        <!-- Nhiệt độ căn giữa -->

        <TextView
            android:id="@+id/tvTemp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="10dp"
            android:layout_marginStart="7dp"
        android:text="25°"
        android:textColor="@android:color/white"
        android:textSize="64sp"
        android:textStyle="bold" />

        <!-- Max/Min ngang hàng -->
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="horizontal"
            android:gravity="center">

            <TextView
                android:id="@+id/tvTempMax"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="↑16°"
                android:textColor="@android:color/white"
                android:textSize="16sp"
                android:layout_marginEnd="24dp" />

            <TextView
                android:id="@+id/tvTempMin"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="↓12°"
                android:textColor="@android:color/white"
                android:textSize="16sp" />
        </LinearLayout>
    </LinearLayout>
```

```
    <!-- Khung dự báo từng giờ -->
    <LinearLayout
        android:id="@+id/hourlyContainer"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        android:background="@drawable/bg_rounded"
        android:padding="16dp"
        android:layout_marginTop="24dp"
        app:layout_constraintTop_toBottomOf="@id/resultContainer"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="parent">

        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/recyclerHourly"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:orientation="horizontal" />
    <!-- Nút xem 5 ngày -->
    <Button
        android:id="@+id/btnFiveDay"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Xem 5 ngày tới"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="16dp" />

    <!-- Khung ẩn hiện dự báo 5 ngày -->



    </LinearLayout>


</androidx.constraintlayout.widget.ConstraintLayout>
```

KẾT LUẬN

1. Kết quả đạt được




18:00
Thời Tiết
Email
Mật khẩu
ĐĂNG NHẬP
ĐĂNG KÝ


1:51
WeatherApp
Thai Binh
broken clouds
29°
↑29° ↓29°
2025-04-11
09:00
28°C
2025-04-11
12:00
25°C
2025-04-11
15:00
23°C
2025-04-11
18:00
23°C
XEM 5 NGÀY TỚI

2. Nhược điểm

Mặc dù ứng dụng "Dự báo thời tiết" đã đáp ứng được các chức năng cơ bản và mang lại trải nghiệm thuận tiện cho người dùng, tuy nhiên trong quá trình xây dựng và vận hành, vẫn còn tồn tại một số nhược điểm và hạn chế sau:

1. Phụ thuộc vào API bên thứ ba
   Ứng dụng sử dụng dữ liệu thời tiết từ dịch vụ công khai như OpenWeatherMap API, vì vậy:
   - Nếu API bị giới hạn truy cập, lỗi hoặc thay đổi cấu trúc dữ liệu, ứng dụng sẽ không thể hoạt động bình thường.
   - Tài khoản miễn phí bị giới hạn số lượng request/ngày, ảnh hưởng khi lượng người dùng tăng cao.
2. Chưa hỗ trợ ngoại tuyến (offline)

- Ứng dụng yêu cầu kết nối Internet để lấy dữ liệu thời tiết mới.
- Trong điều kiện không có mạng, ứng dụng không thể hiển thị thông tin thời tiết hoặc chỉ hiển thị dữ liệu cũ (nếu được cache lại).

3. Giao diện còn đơn giản
    - Dù đã áp dụng Material Design, nhưng một số thành phần giao diện chưa thực sự sinh động hoặc hấp dẫn với người dùng (chưa có animation, hiệu ứng mượt,…).
4. Thiếu tính năng mở rộng nâng cao
    - Ứng dụng hiện tại chỉ tập trung vào thông tin cơ bản như nhiệt độ, độ ẩm, gió,...
    - Chưa hỗ trợ các tiện ích bổ sung như: cảnh báo thời tiết nguy hiểm, biểu đồ trực quan, tích hợp bản đồ mưa, radar,…
5. Chưa tối ưu pin và dữ liệu mạng
    - Việc cập nhật liên tục dữ liệu thời tiết trong khoảng thời gian ngắn có thể gây hao pin hoặc tốn dữ liệu nếu không được tối ưu.
6. Chưa hỗ trợ đa nền tảng
    - Ứng dụng hiện chỉ chạy trên hệ điều hành Android, chưa có phiên bản dành cho iOS hoặc web, do đó giới hạn đối tượng người dùng.

3. Hướng phát triển

Nhằm khắc phục các nhược điểm còn tồn tại và nâng cao chất lượng ứng dụng, trong tương lai nhóm phát triển hướng đến việc cải tiến và mở rộng ứng dụng "Dự báo thời tiết" theo các hướng sau:

1. Nâng cấp giao diện người dùng (UI/UX)
    - Thiết kế lại giao diện với nhiều hiệu ứng chuyển động mượt mà, biểu tượng sinh động và dễ nhận diện.
    - Cập nhật chế độ Dark Mode linh hoạt theo thời gian thực (ngày/đêm).
    - Tối ưu giao diện cho các kích thước màn hình khác nhau (điện thoại, tablet).
2. Hỗ trợ hoạt động ngoại tuyến (Offline Mode)
    - Lưu trữ dữ liệu thời tiết gần nhất bằng Room Database để có thể hiển thị khi mất kết nối mạng.
    - Cho phép người dùng xem lại lịch sử thời tiết đã tra cứu.

3. Thêm tính năng cảnh báo thời tiết khẩn cấp
    - Hiển thị thông báo khi có hiện tượng thời tiết nguy hiểm như: mưa lớn, giông bão, nắng nóng cực đoan,…
    - Đồng bộ với hệ thống cảnh báo quốc gia (nếu có API hỗ trợ).
4. Tích hợp bản đồ và biểu đồ trực quan
    - Sử dụng Google Maps API để hiển thị tình trạng mưa, nhiệt độ theo khu vực.
    - Thêm biểu đồ dự báo nhiệt độ, độ ẩm, áp suất theo giờ và theo ngày.
5. Hỗ trợ đa ngôn ngữ và đa nền tảng
    - Cung cấp giao diện song ngữ (Việt – Anh), có thể mở rộng sang các ngôn ngữ khác.
    - Xây dựng phiên bản ứng dụng cho iOS và web để mở rộng lượng người dùng.
6. Tối ưu hiệu năng và tiết kiệm tài nguyên
    - Giảm tần suất gọi API không cần thiết.
    - Sử dụng kỹ thuật cache thông minh, cập nhật theo thời gian thực một cách linh hoạt.
    - Áp dụng công nghệ Firebase để gửi thông báo, lưu trữ dữ liệu người dùng (nếu cần).
7. Ứng dụng trí tuệ nhân tạo (AI) trong tương lai
    - Phân tích thói quen người dùng để gợi ý thông tin thời tiết phù hợp.
    - Dự đoán xu hướng thời tiết bằng mô hình học máy (machine learning) nếu có đủ dữ liệu.

TÀI LIỆU THAM KHẢO

[1] David Flanagan, JavaScript: The Definitive Guide, 7th Edition, O'Reilly Media, 2020.

[2] Adam Freeman, “Pro jQuery”, Apress, 2018.

[3] Benjamin Jakobus, “Mastering Bootstrap 5”, Packt Publishing, 2018.