Môn học: Lập trình trên thiết bị di động

Mô tả kỹ thuật và nghiệp vụ

Ứng dụng: dự báo thời tiết

Nhóm 22: Phạm Quang Minh, Tất Quảng Kiệt, Lê Công Nhật

1/ Mô tả ứng dụng:

Ứng dụng dự báo thời tiết là ứng dụng thời gian thực được nhóm phát triển để có thể xem thông tin về thời tiết hiện tại và trong 7 ngày tới. Mặc dù còn nhiều chức năng chưa thực hiện được tuy nhiên sản phẩm của nhóm cũng đã khá hoàn thiện về các chức năng chính.

2/ Các dependencies được sử dụng:

* 'com.readystatesoftware.sqliteasset:sqliteassethelper:+' : SqliteAssetHelper để quản lý database
* 'com.google.android.gms:play-services-location:18.0.0': dùng để lấy vị trí của thiết bị
* 'com.squareup.retrofit2:adapter-rxjava2:2.3.0' và 'com.squareup.retrofit2:converter-gson:2.3.0': retrofit2 dùng để kết nối với API
* 'com.squareup.picasso:picasso:2.71828': Picasso dùng để đổ ảnh từ url lên imageView

3/ API

API thời tiết được lấy từ openweathermap.org để sử dụng

4/ Source code:

Kỹ thuật lập trình sử dụng trong ứng dụng bao gồm các kĩ năng lập trình cơ bản và lập trình hướng đối tượng

* MainActivity: - Tại main API sẽ được gọi để hiển thị dữ liệu lên layout mặc định của app đồng thời mở database để tải lên activity các dữ liệu được người dùng thêm vào bằng chức năng add favorite city của app. Hàm generateLayout được truyền vào các thông tin lấy được từ API sau khi chạy hàm getWeatherInfomation.

- Các sự kiện onClick cũng được cài đặt cho layout khi được tạo mới và đưa lên view trong hàm generateLayout

* SearchActivity: - Tại search có 1 thanh AutoComplete TextView được lấy dữ liệu từ database CityFTS.db.

- Database Access sẽ mở database lên và bắt đầu đổ dữ liệu vào AutoCompleteTextView mỗi khi người dùng thay đổi chữ trong TextView

- Sau khi người dùng tìm được thành phố và chọn thì thông tin longtitude và latitude của thành phố sẽ được lưu lại vào bảng save\_favorite\_city sau đó dữ liệu tiếp tục được chuyển về MainActivity để tạo layout và gọi API đổ dữ liệu lên giao diện.

* DetailActivity: -Khi layout được hiển thị lên giao diện cùng dữ liệu được lấy từ API có kèm theo sự kiện onClick khi click vào sẽ chuyển đến DetailActiviy

-Tại DetailActivity dữ liệu về thông tin thành phố của layout được đưa sang và gọi One Call API của OpenWeatherMap để lấy các thông tin của thời tiết hiện tại, 48 giờ tới và 7 ngày tiếp theo.

Hàm tạo mới layout và hiển thị ra view tại MainActivity

public void generateLayout(String temperatureString, String cityName, String path, Coord coord) {  
 final View view = getLayoutInflater().inflate(R.layout.*layout\_add\_city*, null);  
 TextView nhietDo = view.findViewById(R.id.*tv\_temperature*);  
 TextView thanhPho = view.findViewById(R.id.*tv\_city*);  
 ImageView iconThoiTiet = view.findViewById(R.id.*img\_weatherIcon*);  
 arrayList.add(coord);  
 registerForContextMenu(view);  
 nhietDo.setText(temperatureString);  
 thanhPho.setText(cityName);  
 Picasso.*get*().load(path).into(iconThoiTiet);  
  
 view.setOnClickListener(v -> {  
 Intent intent = new Intent(MainActivity.this, DetailActivity.class);  
 intent.putExtra("cityName", cityName);  
 intent.putExtra("lon", coord.getLon());  
 intent.putExtra("lat", coord.getLat());  
 startActivity(intent);  
  
 });  
 view.setTag(arrayList.size());  
 layoutList.addView(view);  
}

Hàm gọi API để lấy dữ liệu thời tiết truyền vào hàm tạo layout tại MainActivity

private void getWeatherInformation(String lat, String lon) {  
 Retrofit retrofit = RetrofitClient.*getInstance*();  
 WeatherService weatherService = retrofit.create(WeatherService.class);  
 Call<WeatherResponse> call = weatherService.getWeatherByLatLon(lat, lon, Common.*API\_KEY\_ID*, "metric");  
 call.enqueue(new Callback<WeatherResponse>() {  
 @Override  
 public void onResponse(@NonNull Call<WeatherResponse> call, @NonNull Response<WeatherResponse> response) {  
 if (response.code() == 200) {  
 WeatherResponse weatherResponse = response.body();  
 assert weatherResponse != null;  
 double temp = weatherResponse.getMain().getTemp();  
 String temperatureString = Double.*toString*(temp) + "°C";  
 String cityName = weatherResponse.getName();  
 String path = "http://openweathermap.org/img/wn/" +  
 weatherResponse.getWeather().get(0).getIcon() +  
 "@2x.png";  
 Coord coo = new Coord(Double.*valueOf*(lon), Double.*valueOf*(lat));  
 generateLayout(temperatureString, cityName, path, coo);  
  
 }  
 }

Sự kiện onClick được set vào layout khi được tạo để chuyển từ MainActivity sang DetailActivity

view.setOnClickListener(v -> {  
 Intent intent = new Intent(MainActivity.this, DetailActivity.class);  
 intent.putExtra("cityName", cityName);  
 intent.putExtra("lon", coord.getLon());  
 intent.putExtra("lat", coord.getLat());  
 startActivity(intent);  
  
});

Khi người dùng chọn addCity trên giao diện sẽ chuyển qua SearchActivity

addCity.setOnClickListener(v -> {  
 Intent intent = new Intent(MainActivity.this, SearchActivity.class);  
 startActivityForResult(intent, *SEND\_CODE*);  
});

Interface được viết để gọi API từ openweather

