# Dokumentacja projektu – Aplikacje mobilne „Przekierowywanie połączeń”

Autorzy:
Wojciech Gałka
Paweł Durda
Informatyka III
gr. Lab III

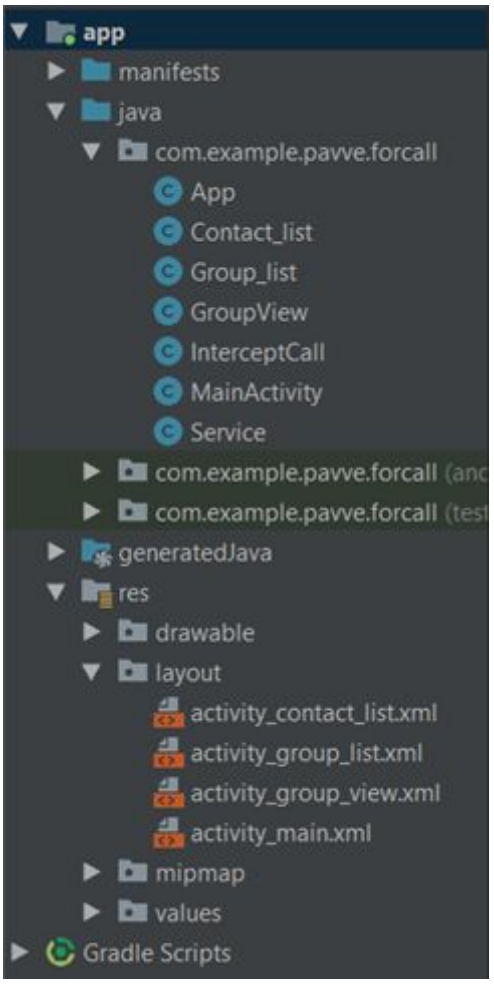
1. Cele projektu

Możliwość włączenia/wyłączenia przekierowania połączenia telefonicznego na wybrany przez użytkownika numer telefonu.

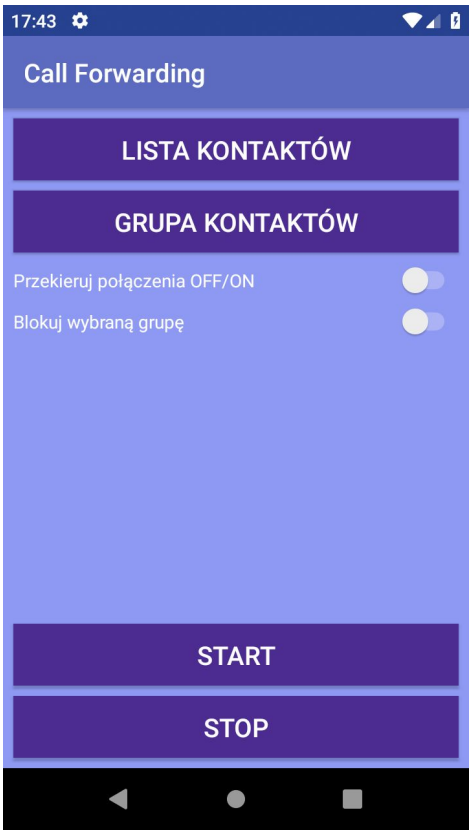
2. Wykorzystane technologie

Aplikacja została napisana w środowisku programistycznym Android Studio w języku Java. Interfejs użytkownika jest zdefiniowany przy użyciu XML. Do przechowywania danych został użyty mechanizm SharedPreferences.

3. Opis projektu

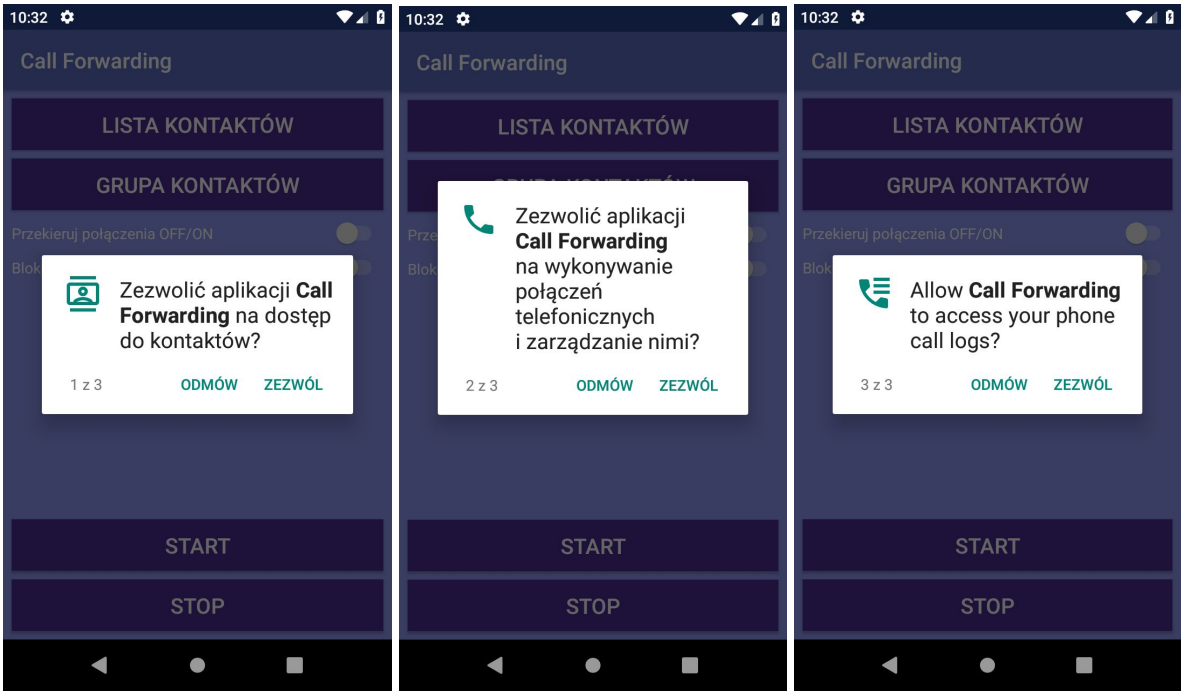



**Widok projektu**

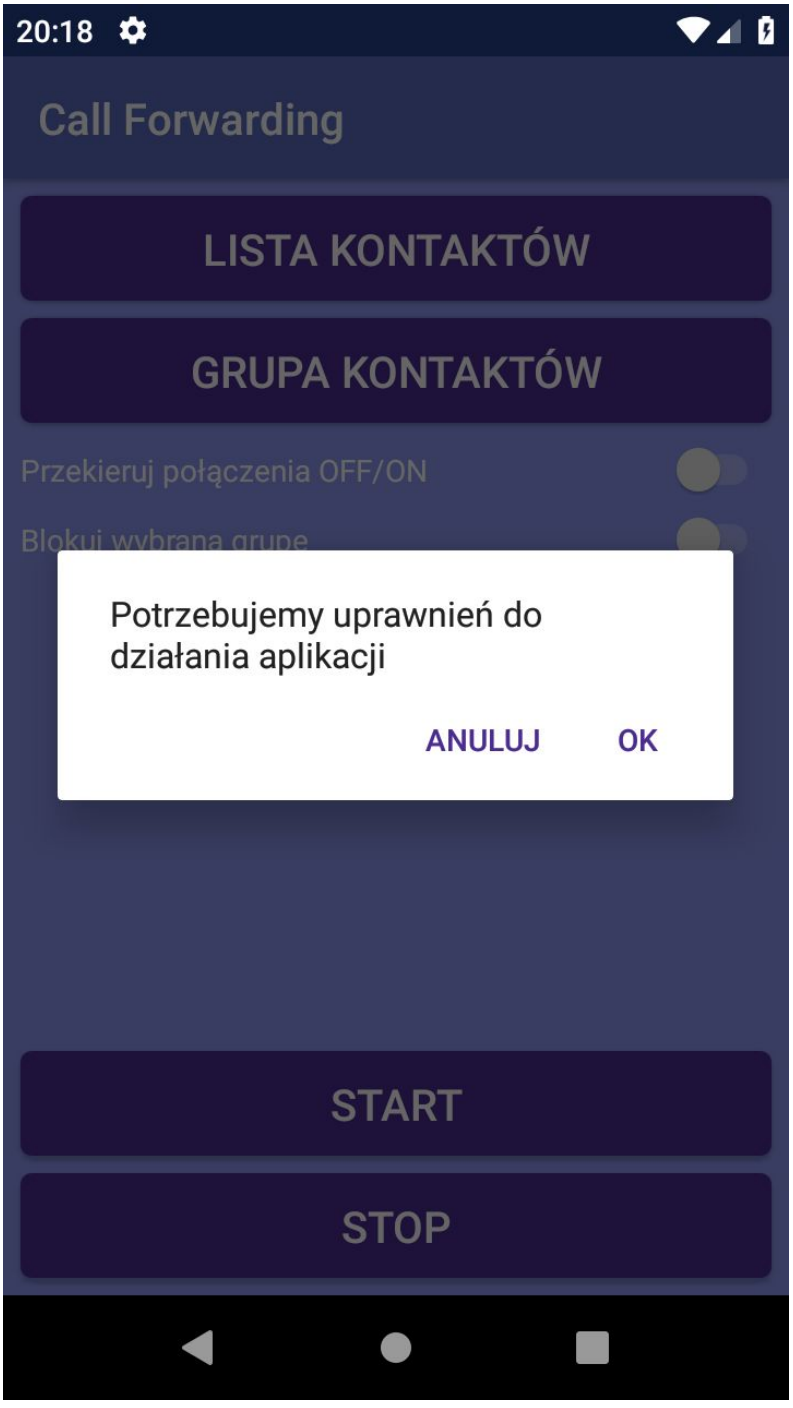



**MainActivity**

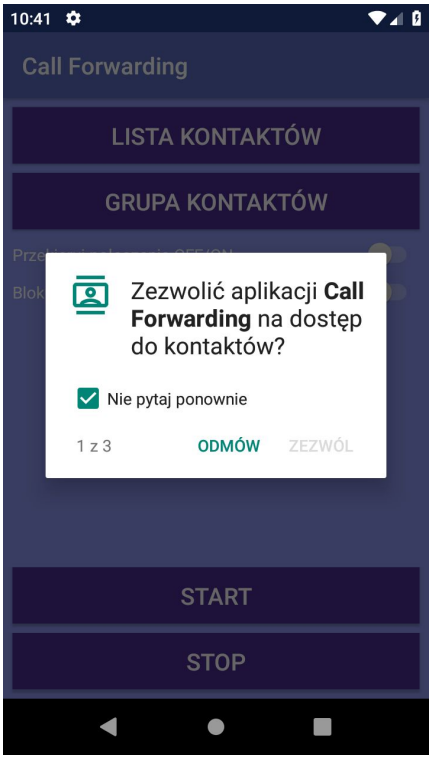
Po uruchomieniu aplikacji użytkownik po wybraniu jednej z opcji która potrzebuje uprawnień do poprawnego działania zostanie poproszony o ich udzielenie.



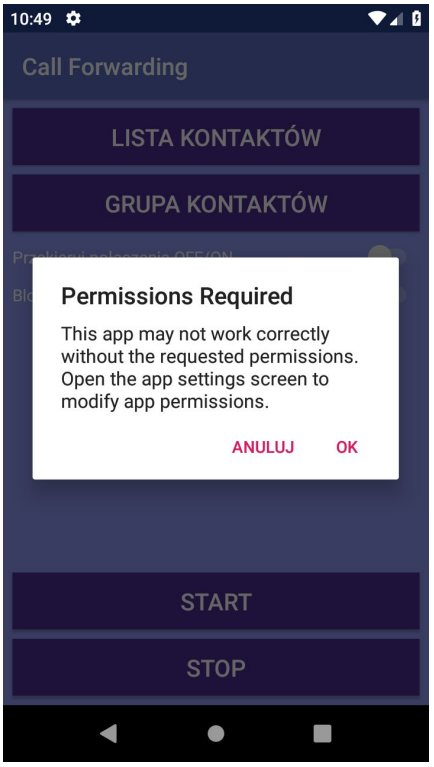
Jeśli użytkownik odmówi udzielenia uprawnień, po czym znów wybierze jedną z opcji zostanie wyświetlony następujący komunikat.



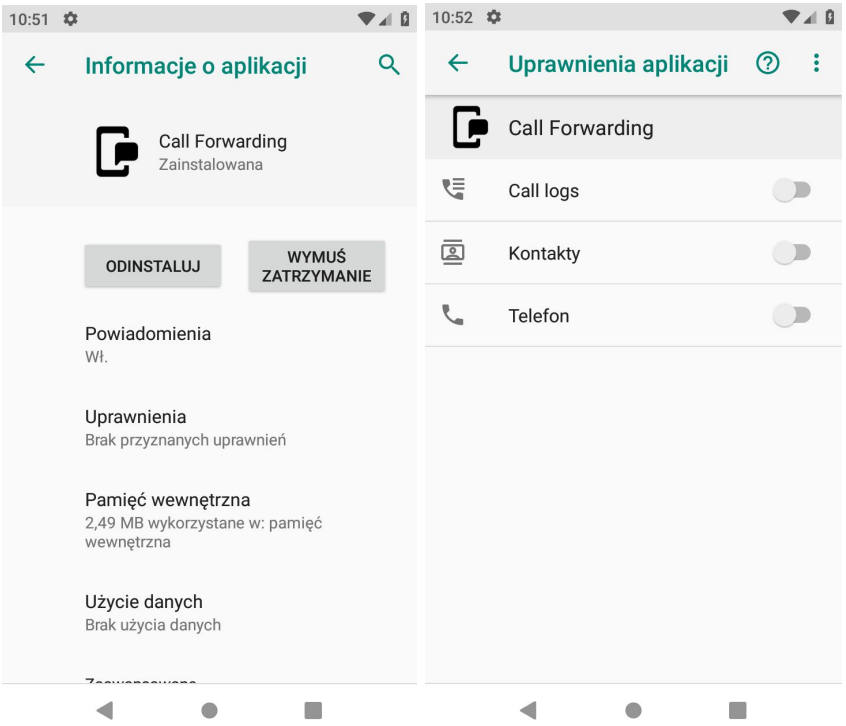

Wybór przycisku “OK” pokaż użytkownikowi okno z prośbą do udzielenia uprawnień.




Jeśli użytkownik w kolejnym podejściu odmówi udzielenia uprawnień zaznaczając “Nie pytaj ponownie” zostanie wyświetlony ostateczny komunikat.




Wybierając opcję “OK” zostaniemy przekierowani do ustawień aplikacji gdzie użytkownik musi ręcznie przydzielić uprawnień.

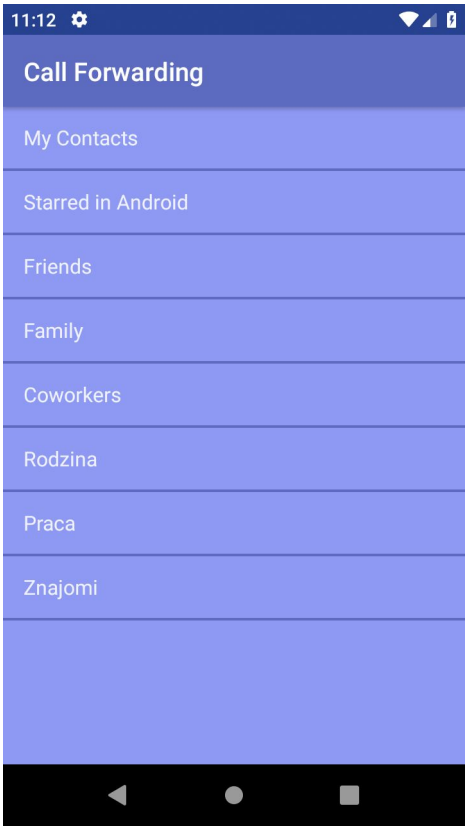



Użytkownik udzielając potrzebnych uprawnień uzyskuje dostęp do aplikacji, możliwość wyboru numeru telefonu na który zostaną przekierowane połączenia przychodzące.
Klikając w przycisk „Lista kontaktów” zostajemy przeniesieni do widoku „Contact_list”, gdzie wybieramy kontakt, z którego zostanie pobrany numer telefonu.


**Contact_list**

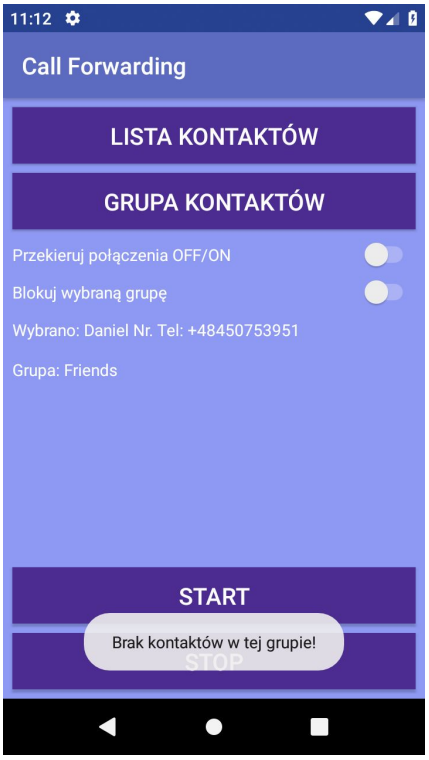
Po wyborze kontaktu następuje powrót do widoku MainActivity, gdzie widoczny jest wybrany przez nas numer telefonu, na który będą przekierowywane połączenia.


Opcja “Grupa kontaktów” pokazuje grupy kontaktów. Użytkownik wybiera jedną z nich.

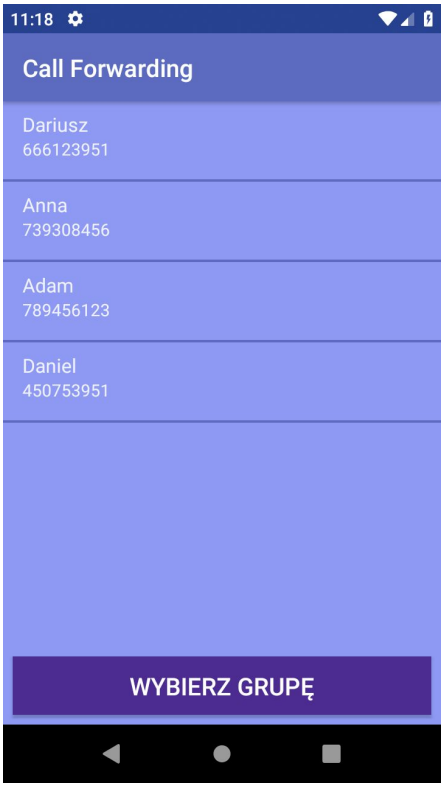



**Group_list**

Jeśli zostanie wybrana grupa, która nie posiada ani jednego kontaktu, następuje powrót do MainActivity oraz zostaje wyświetlony komunikat.

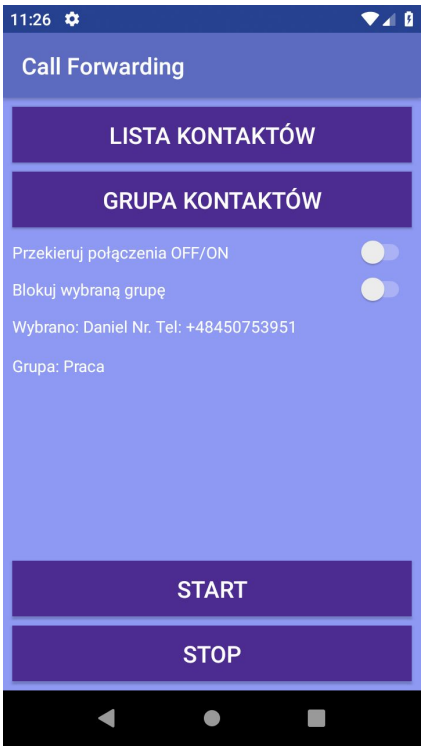



Jeśli użytkownik wskaże grupę w której jest przynajmniej jeden kontakt, zostanie przeniesiony do widoku “Group_view”, gdzie widać wybraną grupę kontaktów oraz przycisk potwierdzający “Wybierz grupę”.

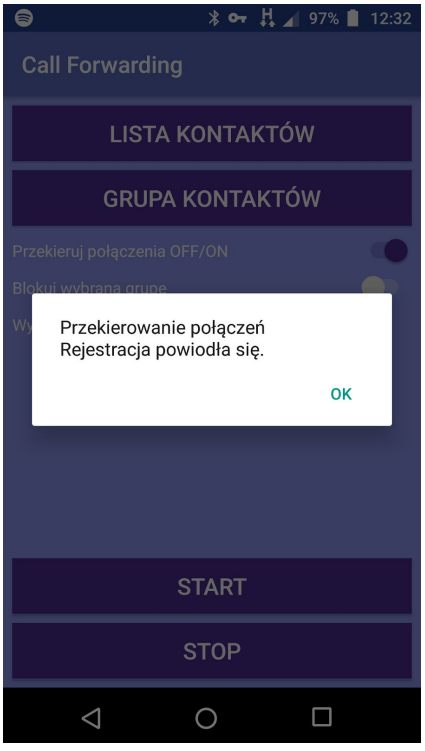



**Group_view**

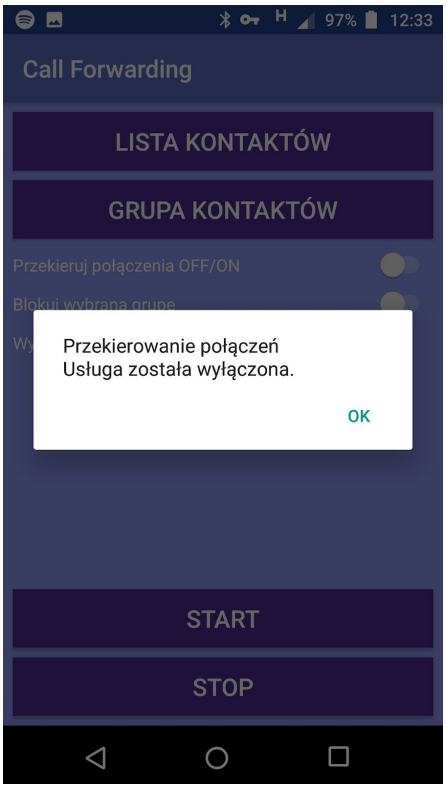
Po wyborze grupy kontaktów następuje powrót do widoku MainActivity.




Przełącznik “Przekieruj połączenia OFF/ON” można aktywować w momencie, gdy jest wybrany numer telefonu z widoku “Lista kontaktów”, również jego stan jest zapisany przez mechanizm SharedPreferences. Po zamknięciu aplikacji i jej ponownym uruchomieniu jest możliwość odwołania przekierowania połączenia u operatora.

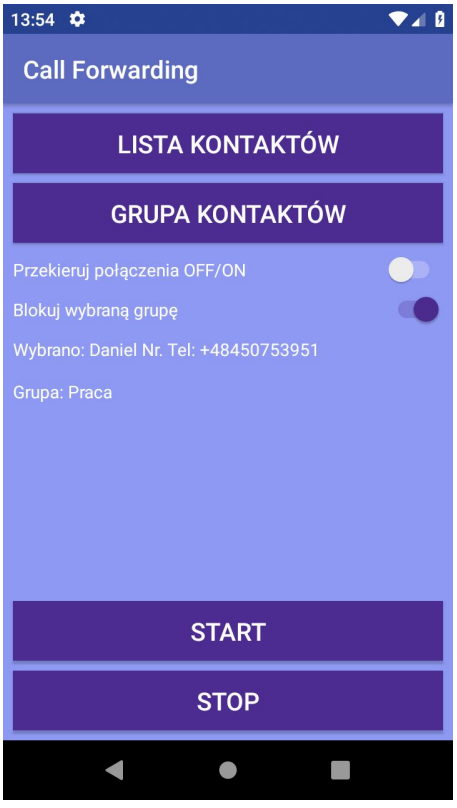



*Wynik wykonanej operacji. Aktywacja przekierowania połączenia.*




*Wynik wykonanej operacji. Dezaktywacja przekierowania połączenia.*

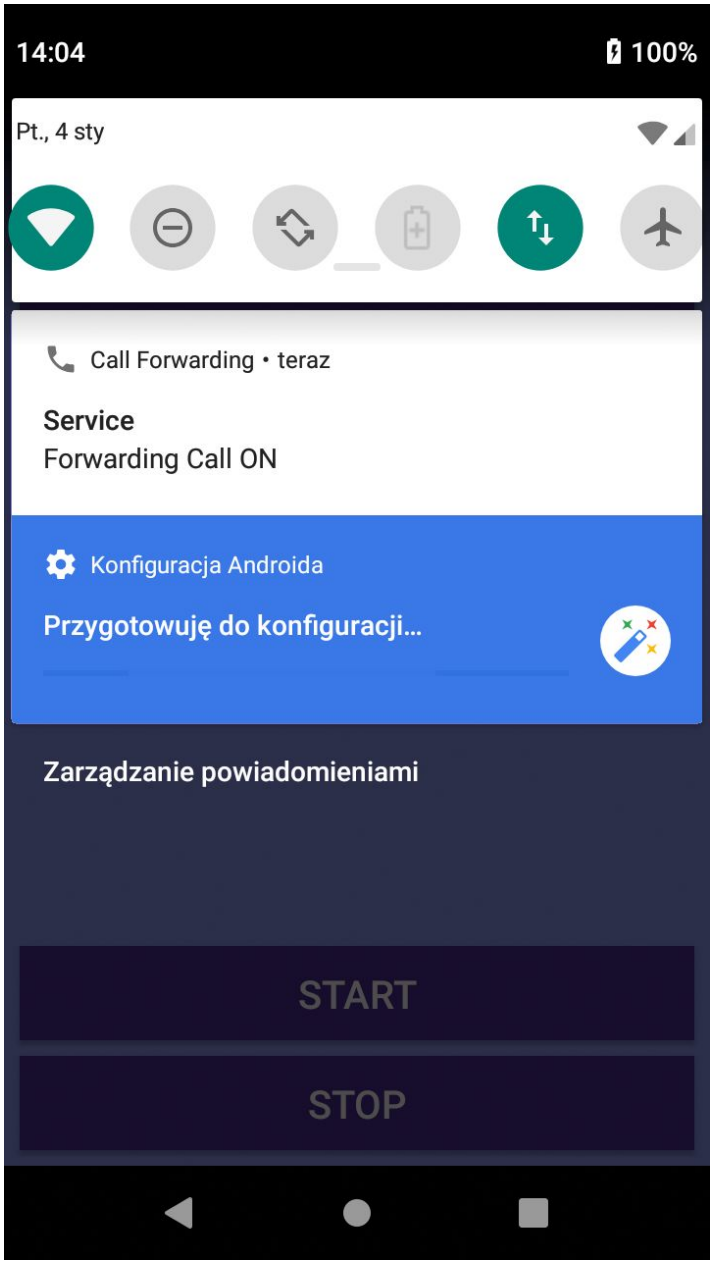
Przełącznik “Blokuj wybraną grupę” można aktywować w momencie, gdy jest wybrana grupa kontaktów z widoku “Grupa kontaktów”.



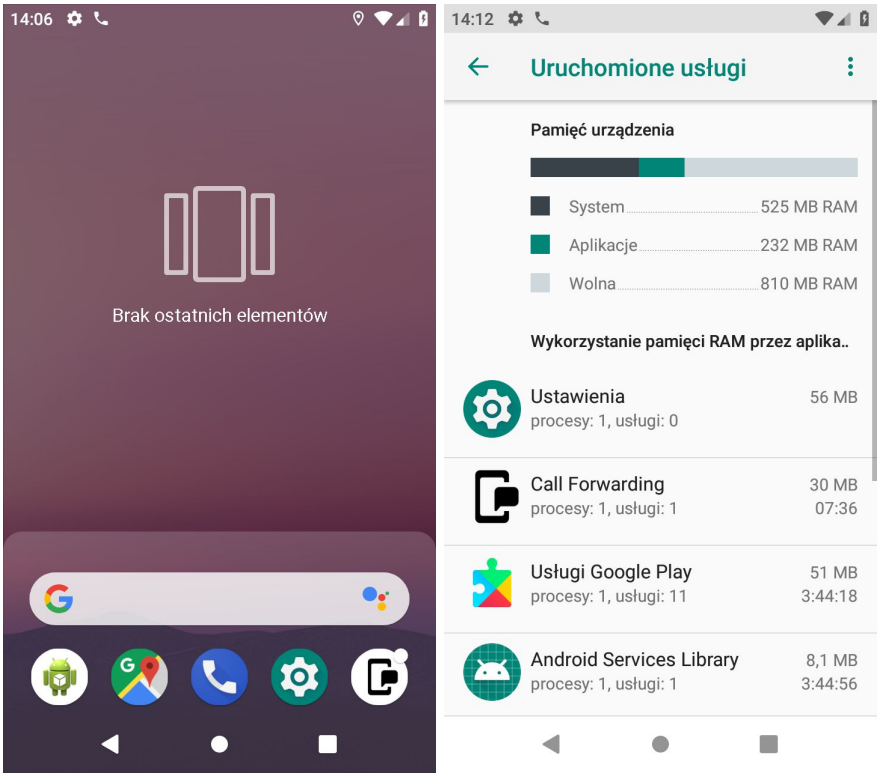

Jeśli opcja blokady wybranej grupy jest aktywna, połączenia przychodzące będą odrzucane.

Jeśli użytkownik również ma wybraną opcję “Przekieruj połączenia OFF/ON”, połączenie przychodzące z wybranej grupy zostanie odrzucone, a następnie przekierowane na wybrany wcześniej numer telefonu.

Przycisk “Start” oraz “Stop” odpowiadają za włączenie/wyłączenie działania aplikacji w tle.
Po aktywacji jest pokazane powiadomienie, aby je zobaczyć należy rozsunąć “belkę” powiadomień.



Po wyłączeniu aplikacji jest proces działa w tle. Jeżeli są uruchomione usługi przekierowanie połączenia lub blokada połączeń przychodzących od wybranej grupy kontaktów, dalej spełnia ona swoje zadanie.




Klikając w powiadomienie powracamy do widoku MainActivity.

4. Opis fragmentów kodu

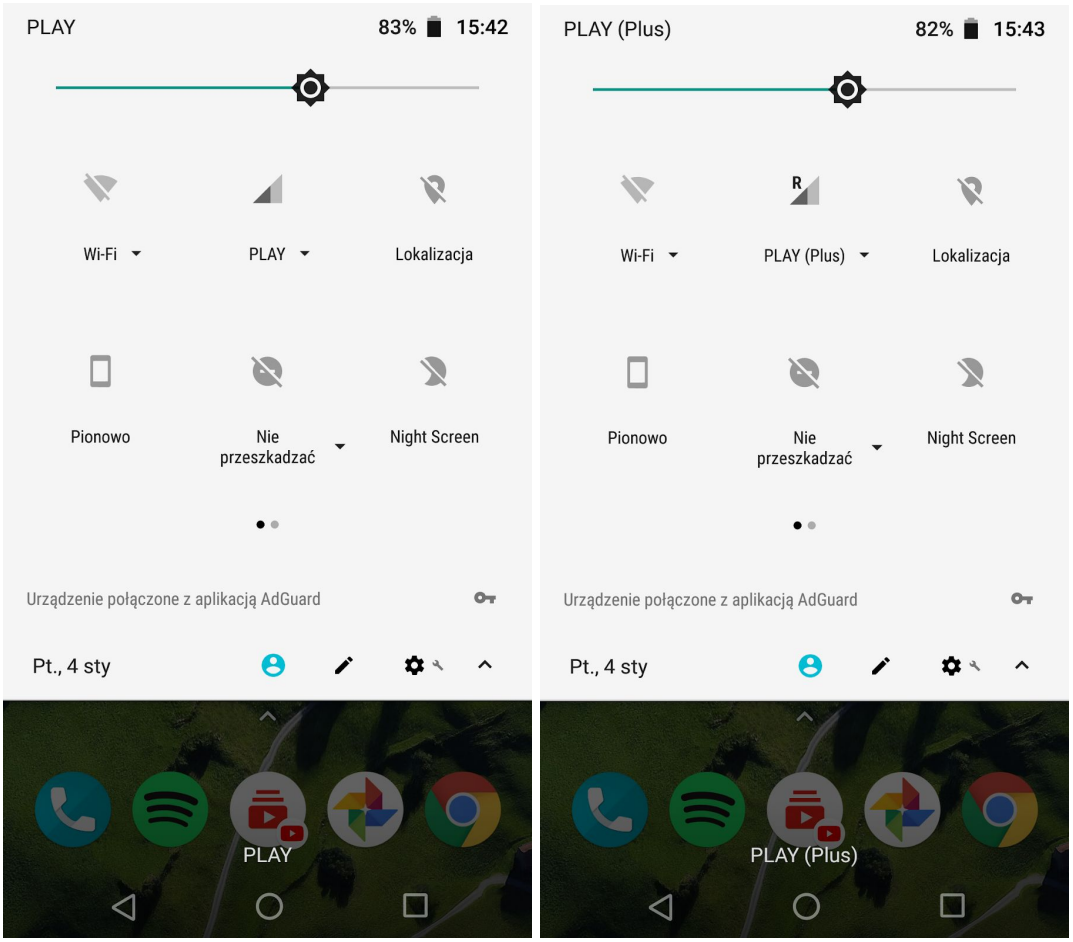
Chcą uprościć użytkownikowi korzystanie z aplikacji do wyboru kontaktu oraz grupy kontaktów, aplikacja odczytuje nazwę operatora i do niego dobiera odpowiednie kody MMI. Użytkownik dzięki temu nie musi pamiętać kodów MMI, aby włączyć lub wyłączyć usługę przekierowania połączenia.

```
public void acceptedNumer(){
    TelephonyManager telephonyManager = (TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
    String simOperatorName = telephonyManager.getNetworkOperatorName();

    String[] parts = simOperatorName.split( regex: " ");
    simOperatorName = parts[0];
    finalSimOperatorName = simOperatorName;

    switch(simOperatorName) {
        case "PLAY":
            gdy_zajety = "*67*";
            ODgdy_zajety = "%2367%23";
            break;
        case "Orange":
            gdy_zajety = "**67*";
            ODgdy_zajety = "%2367%23";
            break;
        case "PLUS":
            gdy_zajety = "*67*";
            ODgdy_zajety = "%2367%23";
            break;
        case "T-Mobile.pl":
            gdy_zajety = "*67*";
            ODgdy_zajety = "%2367%23";
            break;
        case "nju":
            gdy_zajety = "**67*";
            ODgdy_zajety = "%2367%23";
            break;
        case "Android": //emulator
            gdy_zajety = "";
            ODgdy_zajety = "";
            break;
        default: //Operator nieznany
            finalSimOperatorName = null;
    }
```

*Metoda odpowiedzialna za odczytanie nazwy operatora i dobranie do niego kodów MMI.*




Wyżej wymienione przykłady pokazują,pomimo tego że użytkownik znajduje się w sieci “PLAY”, korzysta z nadajników sieci “Plus”. Kody MMI u tych operatorów mogą się różnić, więc łańcuch znaków w tym przypadku “PLAY (Plus)” zostanie podzielony, tak aby uzyskać nazwę sieci macierzystej “PLAY”. W ten sposób użytkownikowi są dobierane odpowiednie kody MMI.

Wykonanie połączenia, aby włączyć usługę przekierowania połączenia.

***67*(numer_telefonu)#**

*Przykład*

Kod MMI poprzedzający numer telefonu, na który przekierowane zostaną połączenia
# - znak kończący

W specyfikacji URI, aby zastosować znak “#”, aby poprawnie wykonać “Intent.ACTION_CALL” należy zastąpić go “%23”.

```
Uri call = Uri.parse("tel: " + gdy_zajety + phoneNumber + "%23");
saveData();
Intent dialIntent = new Intent(Intent.ACTION_CALL, call);
startActivity(dialIntent);
```

```java
public class InterceptCall extends BroadcastReceiver {

    @RequiresApi(api = Build.VERSION_CODES.P)
    @Override
    public void onReceive(Context context, Intent intent) {
        try{
            String state = intent.getStringExtra(TelephonyManager.EXTRA_STATE);
            if (state.equalsIgnoreCase(TelephonyManager.EXTRA_STATE_RINGING)) {
                String numer = intent.getStringExtra(TelephonyManager.EXTRA_INCOMING_NUMBER);
                Toast.makeText(context, text: "Dzwoni "+numer,Toast.LENGTH_SHORT).show();
                if(numer.startsWith("+"))
                {
                    numer = numer.substring(3);
                }
                TelecomManager tm = (TelecomManager) context.getSystemService(Context.TELECOM_SERVICE);
                if(MainActivity.grupaNumerowBlokowanych.isEmpty() == false)
                {
                    for(int i=0;i<MainActivity.grupaNumerowBlokowanych.size();i++)
                    {
                        if(MainActivity.grupaNumerowBlokowanych.get(i).equals(numer))
                        {
                            if (tm != null && MainActivity.callBlocker.isChecked()) {
                                boolean success = tm.endCall();
                            }
                        }
                    }
                }
            }
            if (state.equalsIgnoreCase(TelephonyManager.EXTRA_STATE_OFFHOOK)) {
                String numer = intent.getStringExtra(TelephonyManager.EXTRA_INCOMING_NUMBER);
                Toast.makeText(context, text: "Połączenie z "+numer,Toast.LENGTH_SHORT).show();
            }
            if (state.equalsIgnoreCase(TelephonyManager.EXTRA_STATE_IDLE)) {
                Toast.makeText(context, text: "IDLE",Toast.LENGTH_SHORT).show();
            }
        } catch (Exception  e){
            e.printStackTrace();
        }
    }
}
```

**InterceptCall**

Klasa odpowiedzialna za nasłuchiwanie połączeń telefonicznych. Jej głównym zadaniem jest zablokowanie połączenia przychodzącego.

*TelecomManager tm = (TelecomManager) context.getSystemService(Context.TELECOM_SERVICE);*
*boolean success = tm.endCall();*