UNIWERSYTET GDAŃSKI
WYDZIAŁ MATEMATYKI, FIZYKI I INFORMATYKI

Tomasz Wilk 215549

Kierunek: Informatyka
Specjalność: Aplikacje internetowe i bazy danych
Rodzaj studiów: zaoczne

Aplikacja RemembrCall

Współtwórcy: Jessica Tkacz 215561

Praca licencjacka napisana
pod kierunkiem dr Włodzimierza Bzyla

Gdańsk 2016

OŚWIADCZENIE

Ja, niżej podpisana, oświadczam, że przedłożona praca dyplomowa (będąca rezultatem pracy zespołu studentów) została przygotowana przeze mnie – w części, za którą odpowiadam - samodzielnie i nie narusza praw autorskich, interesów prawnych oraz materialnych innych osób.

Ponadto oświadczam, że złożona wersja drukowana i elektroniczna niniejszej pracy licencjackiej są ze sobą zgodne.

…….. ……..

data podpis

Spis treści

1. Streszczenie

Jako pracę licencjacką stworzyliśmy aplikację dla systemu Android o nazwie RemembrCall, której zadaniem jest przypominać użytkownikowi o wykonaniu telefonu do wybranego kontaktu. Naszym podstawowym celem jest wspomóc zapominalską lub zapracowaną osobę w relacjach z rodziną. Pomysł powstał na podstawie własnych doświadczeń oraz w wyniku braku satysfakcjonującego nas rozwiązania na rynku.

Nazwa aplikacji dość bezpośrednio wskazuje na jej funkcjonalność, lecz kryje się w niej również nawiązanie do dobrze większości ludzi znanej sagi o Harrym Potterze. Kolega Harry,ego, Neville, był bardzo zapominalskim chłopcem, dlatego babcia przysłała mu Przypominajkę (ang. Remembrall), która zmieniała kolor za każdym razem, kiedy Neville o czymś zapomniał. Nasza aplikacja RemembrCall jest więc kombinacją angielskiej nazwy Przypominajki oraz *Call* (dzwonić), co niebanalnie wskazuje sposób na jej zastosowanie.

Ze względu na fakt, iż na rynku smartfonów królują w większości te zasilane systemem Android, co bezpośrednio przekłada się na liczbę odbiorców, zdecydowaliśmy się stworzyć naszą aplikację właśnie dla tego systemu. Dokonaliśmy tego z wykorzystaniem środowiska Android Studio, posługując się językiem programowania Java. Wersja załączona do pracy operuje na systemie minutowym i w takiej postaci była również testowana, natomiast w oficjalnej wersji zamiast minut wprowadziliśmy dni.

W czasach, gdzie praktycznie każdy ma do dyspozycji smartfona bardzo zależało nam, by nasza aplikacja była prosta w obsłudze, żeby mogły z niej korzystać osoby w różnym wieku. Z tego powodu stworzony interfejs jest przejrzysty i nieskomplikowany, tak by nie nastręczał problemów osobie mało obeznanej z coraz to nowszymi technologiami. Naszym zamysłem było, by od strony użytkownika aplikacja była mało wymagająca, dlatego właściwie jedyną czynnością, jaką musi on wykonać to wybrać częstotliwość połączeń z wybranym kontaktem. Doszliśmy również do wniosku, że nie chcemy tworzyć kolejnej aplikacji z niepotrzebnymi funkcjami i częstymi aktualizacjami, które wzbudzają jedynie irytację, gdyż takich jest na rynku do wyboru aż nadmiar. Nasza

RemembrCall spełnia swe zadanie i działa tak jak powinna, a przy tym nie wymaga wiele czasu ani specjalnych umiejętności.

Napisana przez nas aplikacja RemembrCall dostępna jest bezpłatnie na platformie Sklep Play, która funkcjonuje na wszystkich telefonach oraz tabletach posiadających system operacyjny Android. Ponadto wykonany i udostępniony w internecie został również krótki filmik pt. "RemembrCall" opowiadający w skrócie o działaniu naszej aplikacji. Został również załączony do niniejszej pracy w wersji elektronicznej.

2. Wstęp

Wartości rodzinne i ogólnie międzyludzkie są jednymi z najważniejszych dla każdego człowieka. W obecnych czasach mamy bardzo rozwiniętą sieć komunikacji, nie musimy pisać listów i czekać tygodniami na odpowiedź. Wystarczy sięgnąć po telefon bądź komunikator internetowy i możemy porozmawiać nawet twarzą w twarz z osoba będącą w dowolnym miejscu na świecie. Paradoksalnie urządzenia, które miały pomóc w tym kontakcie coraz częściej oferują inne zastosowania, pochłaniające czas i kierujące myśli na inne tory, także często okazuje się, że zapominamy o tym, co najważniejsze. Również z większą ilością obowiązków, pracą, zatracamy lub tłumimy tę potrzebę kontaktu z najbliższymi. Bardzo wielu młodych ludzi zapomina jak ważne są rozmowy czy spotkania dla starszego pokolenia. Tymczasem wystarczy raz na tydzień czy dwa zadzwonić do babci, by była szczęśliwa. Dzięki temu i ta młoda osoba, która być może na razie nie widzi takiej potrzeby, na przyszłość nie będzie żałowała zaniedbywania tej sfery życia. Inaczej w efekcie nadchodzi moment, kiedy człowieka nachodzi refleksja, że chciałby to zmienić,a czasem bywa za późno.

Powodem, dla którego podjęliśmy ten temat jest fakt, iż sami znajdujemy się w grupie potencjalnych odbiorców. Po sobie jaki i po naszych kolegach widzimy, iż relacje rodzinne ulegają stopniowo rozluźnieniu. Jest to problem zwłaszcza teraz, w czasach, gdy ludzie są rozpraszani wieloma innymi zajęciami. Zdarza się, że rodzice muszą prosić dorosłe dziecko by zadzwoniło do dziadków, żeby nie było im przykro, co jest żenujące dla obu stron. A przecież to wcale nie jest tak, że nie chcemy albo nas to nie obchodzi. Zwyczajnie ta sprawa zanika wśród wielu innych, zapomina się, przekłada na później. Stworzona przez nas aplikacja RemembrCall ma na celu wspomóc pielęgnowanie więzi poprzez choćby cotygodniową rozmowę.

Istnieją oczywiście aplikacje dla systemu Android, które choć po części miały spełniać podobną rolę, ale posiadały one zazwyczaj zbyt wiele skomplikowanych ustawień oraz niepotrzebnych funkcji, do których po krótkim czasie traciło się cierpliwość. Ponadto aplikacja wszystkim swoim użytkownikom wysyła adnotacje z przypomnieniem o np. Dniu Dziadka albo Dniu Matki, co jest ponadprogramową okazją do wykonania telefonu.

3. Analiza problemu

3.1. Porównanie dostępnych rozwiązań

Analiza dostępnych rozwiązań rozpoczęła się na długo przed podjęciem decyzji o pisaniu naszej aplikacji, jako że szukaliśmy czegoś podobnego do własnego użytku.

3.1.1. Call Reminder (*Duckbone Apps*)

Aplikacja dostępna w Sklepie Play na platformę Android. Nazwa sugeruje funkcjonalność zbliżoną do RemembrCall.

Podobieństwa: Jest możliwość ustawienia przypomnienia o wykonaniu telefonu do konkretnego numeru o konkretnej godzinie, można również ustawić częstotliwość przypomnienia najrzadziej na co tydzień.

Różnice: Oprócz przypomnieniu o telefonie dostępne jest również przypomnienie o wysłaniu sms'a, maila, dodanie notatki. Występuje konieczność częstego powracania do ustawień. W przeciwieństwie do RemembrCall brak możliwości obsługi w języku polskim. Interfejs oraz sposób dokonywania ustawień bardziej złożone i kłopotliwe dla starszych i mniej zorientowanych technologicznie odbiorców. Wymaga nadania większej ilości uprawnień. Posiada reklamy.

3.1.2. Call Reminder (AppAspect Technologies Pvt. Ltd.)

Kolejna aplikacja o tej samej nazwie, jak i zastosowaniu, również przeznaczona dla użytkowników systemu Android.

Podobieństwa: Głównym zadaniem obu aplikacji jest przypomnienie użytkownikowi o wykonaniu telefonu. Obie również mają możliwość wyboru częstotliwości połączeń

Różnice: Trzeba manualnie wprowadzić numer oraz nazwę kontaktu, można wybrać godzinę przypomnienia oraz częstotliwość przypomnień nawet raz na rok. Do wyboru również przypomnienie o wysłaniu sms'a oraz możliwość zrobienia notatki. Aplikacja niepotrzebnie wyświetla u siebie spis połączeń oraz książkę kontaktów. Call Reminder dokonuje przypomnienia w sposób dość

inwazyjny tj. wyświetla powiadomienie na środku ekranu wydając przy tym irytujący dźwięk, trzeba zadzwonić natychmiast albo przypomnienie się usuwa. RemembrCall wysyła przypomnienie dyskretnie i pozostaje na górze ekranu do czasu, aż użytkownik zdecyduje się je wybrać.

3.2. Możliwości zastosowania praktycznego

Aplikacja ma na celu wspomóc osoby, które z różnych powodów nie pamiętają o wykonaniu telefonu do bliskich osób, choć ważne jest dla nich utrzymywanie dobrych stosunków z nimi oraz regularny kontakt. Naszym celem było by prosty w obsłudze interfejs oraz jednorazowa konieczność tworzenia ustawień sprawiły by aplikacja mogła cieszyć się popularnością wśród różnych grup wiekowych. Nasze rozwiązanie może posłużyć zarówno młodzieży, która przez nawał obowiązków nie zawsze pamięta o tym, żeby zadzwonić do ukochanej babci, jak i również osobom starszym w kontaktach z rodziną czy w przypadku regularnych wizyt u lekarza wymagających rejestracji. Ponadto aplikacja może zostać wykorzystana przez specjalistów różnych dziedzin jako wsparcie w kontakcie z klientami w regularnych odstępach czasu (np. comiesięczne przypomnienie o wizycie ortodontycznej).

4. Wymagania funkcjonalne i niefunkcjonalne

4.1. Wymagania funkcjonalne

Aplikacja pobiera pełną listę kontaktów z telefonu użytkownika i wyświetla ją wraz z opcjami ustawień. Właściciel ustawia pożądaną częstotliwość przypomnień o połączeniu z danym kontaktem za pomocą *seekBar*, a obok prezentują się rezultaty wykonanych czynności w postaci ilości dni. Następnie wstępnie zapisuje te ustawienia za pomocą *checkBox*, ponieważ tylko zaznaczone w ten sposób kontakty zostaną uwzględnione przez aplikację. Jest to zabezpieczenie na wypadek, gdyby użytkownik przypadkiem ustalił priorytet w kontakcie, na temat którego powiadomień nie chce otrzymywać. Na sam koniec wystarczy potwierdzić wszystkie ustawienia za pomocą guzika na dole, co da aplikacji ostateczną wersję, którą ma wziąć pod uwagę. W każdym momencie użytkownik może powrócić do owych ustawień i je zmienić.

4.2. Wymagania niefunkcjonalne

Oczywistym ograniczeniem jest fakt, iż aplikacja jest zorientowana na system Android i na żadnym innym nie będzie działała. Możliwe jest używanie jej zarówno na smartfonach, jak i tabletach posiadających wersję systemu Android 5.0. i wzwyż. Można również uruchomić aplikację w środowisku Android Studio, w którym była pisana i oglądać efekty próbnych działań na wybranym emulatorze posiadającym odpowiednią wersję systemu.

5. Projekt systemu

5.1. Architektura rozwiązania

5.1.1. Przypadki użycia

UC-1: Uruchomienie aplikacji

Główny scenariusz:

1. Użytkownik otrzymuje listę kontaktów

2. Użytkownik znajduje wybrany kontakt

3. Użytkownik ustala pożądaną częstotliwość kontaktu z wybraną osobą używając paska przewijania

4. Użytkownik potwierdza wybór zaznaczając pole typu Checkbox

5. Użytkownik potwierdza ustawienia przyciskiem Zapisz.

UC-2: Dodanie/Usunięcie kontaktu

Główny scenariusz:

1. Użytkownik dodaje kontakt w swojej książce adresowej(Kontakty)

2. Użytkownik znajduje dodany/edytowany kontakt na końcu listy kontaktów w aplikacji

    1.2. Użytkownik usuwa kontakt w swojej książce adresowej

    1.3. Użytkownik edytuje kontakt w swojej książce adresowej.

UC-3: Ukrycie aplikacji

Główny scenariusz:

1. Użytkownik przechodzi z aplikacji do pulpitu/wychodzi z aplikacji/zostawia aplikację działającą w tle.

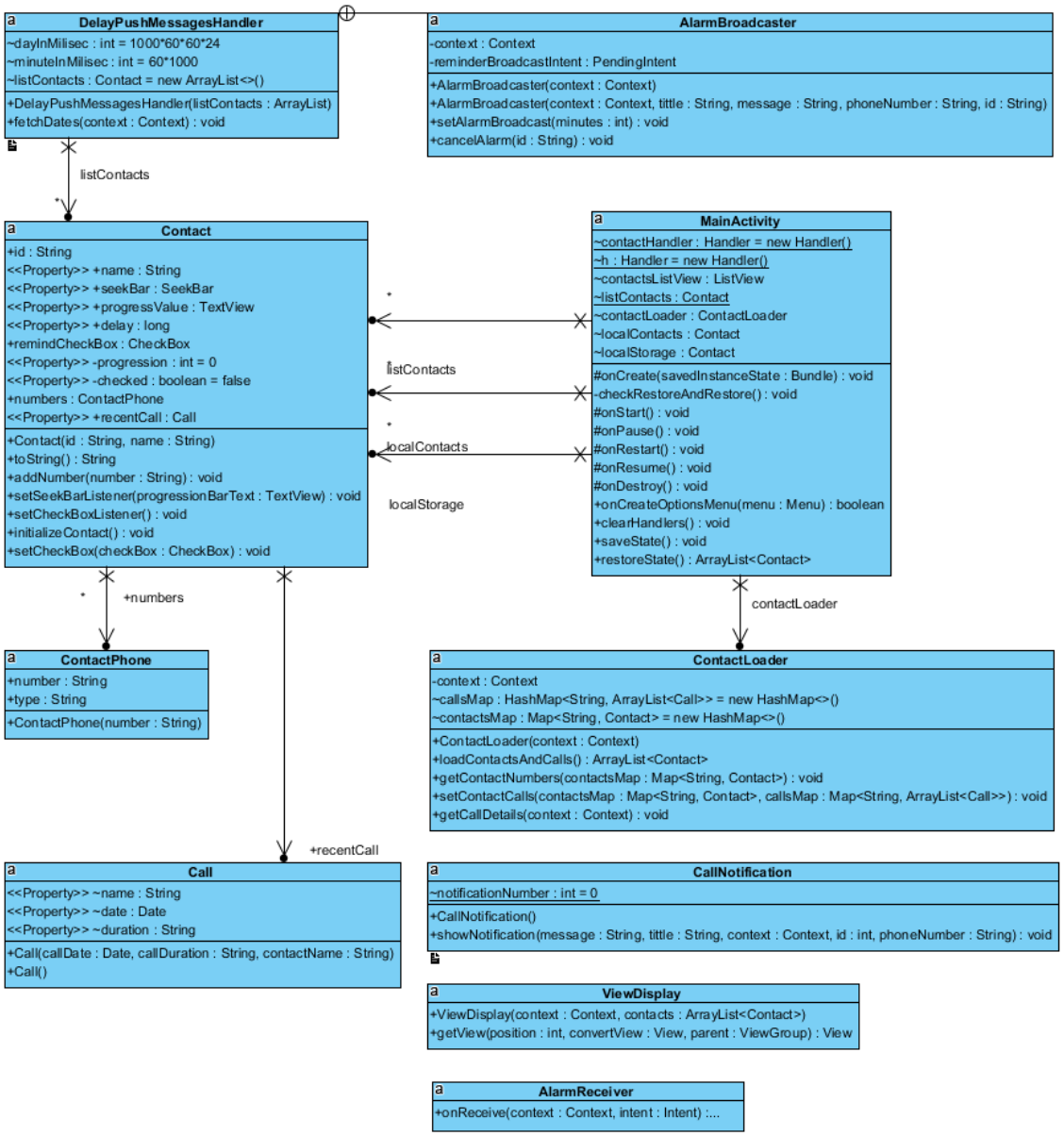
UC-4: Naciśnięcie notyfikacji Główny scenariusz:

1. Użytkownik rozwija pasek notyfikacji

2. Użytkownik dotyka wybraną notyfikację

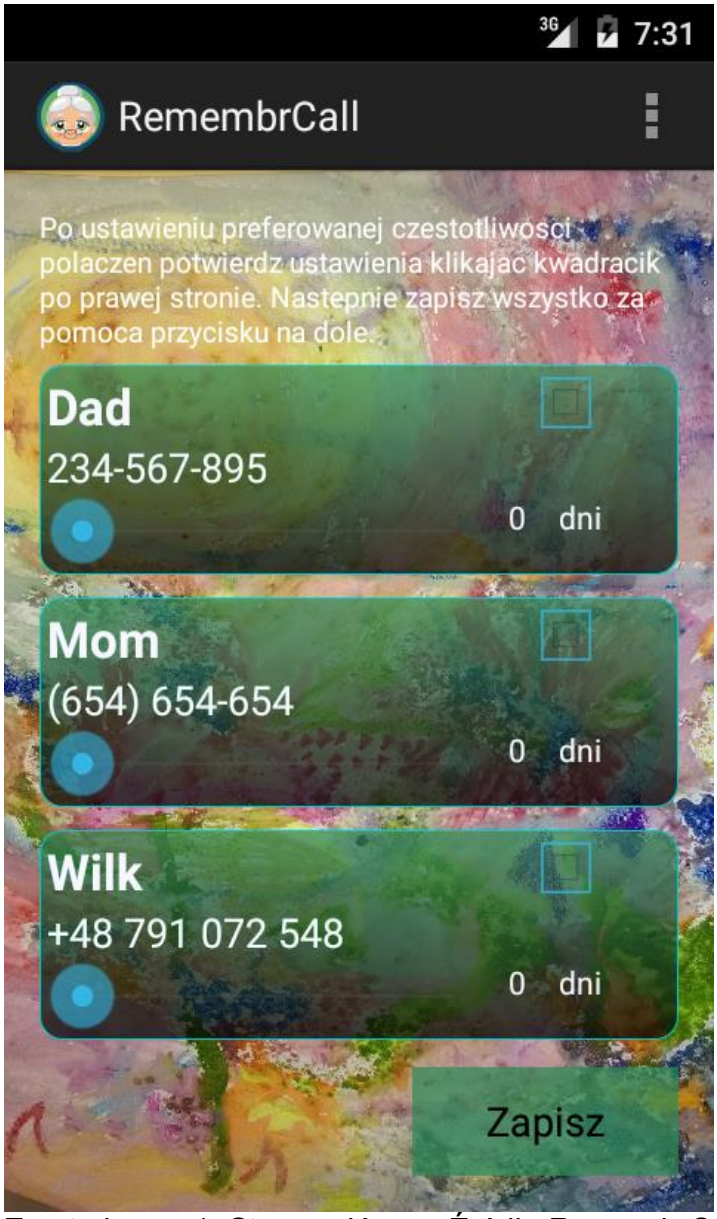
3. Użytkownik przechodzi z notyfikacji do aplikacji Kontakty/Telefon z wybranym numerem

5.1.2. Diagram klas



5.2. Projekt interfejsu użytkownika

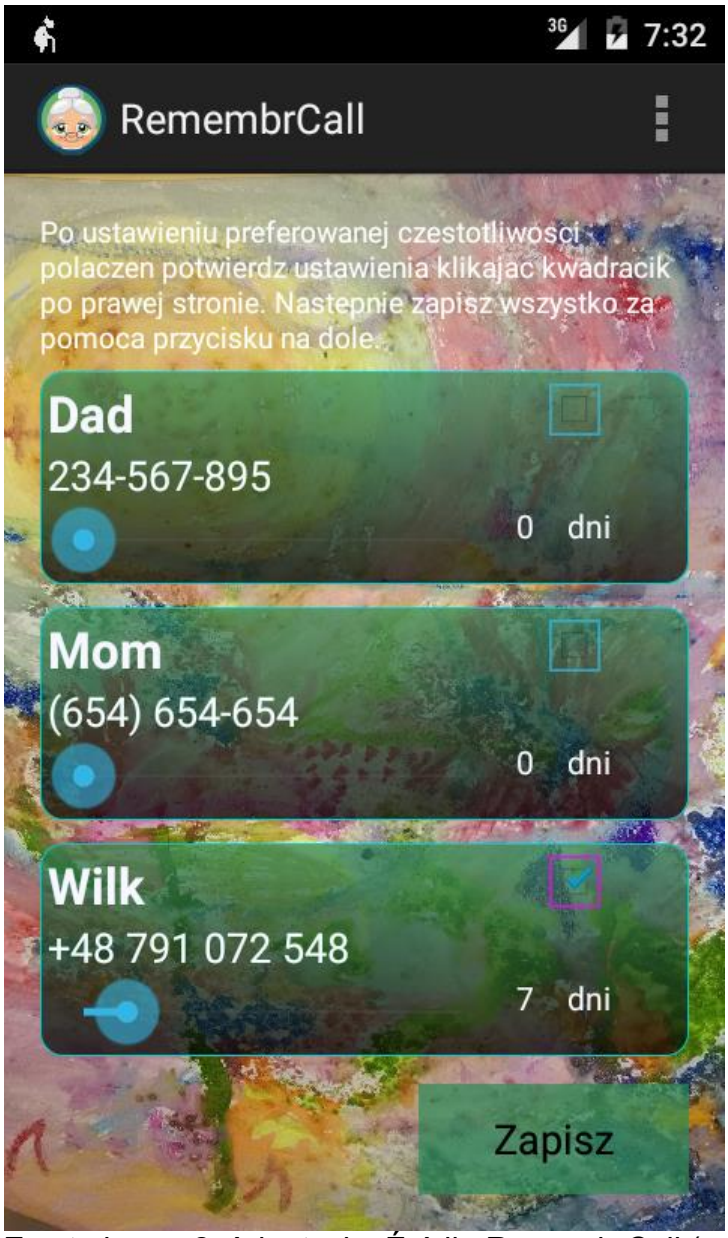
a)



Zrzut ekranu:1: Strona główna; Źródło:RemembrCall (opracowanie własne)

Strona główna po uruchomieniu aplikacji RemembrCall. Wyświetlają się tutaj wszystkie kontakty pobrane z telefonu. Do każdego kontaktu widzianego w osobnej ramce mamy opcje wyboru częstotliwości połączeń oraz pole do zaznaczenia aby ustawienia zostały wzięte pod uwagę. Na dole po prawej stronie widnieje przycisk „Zapisz”, którego wciśnięcie jest konieczne do działania aplikacji. Na samej górze znajduje się krótki opis korzystania z niej.

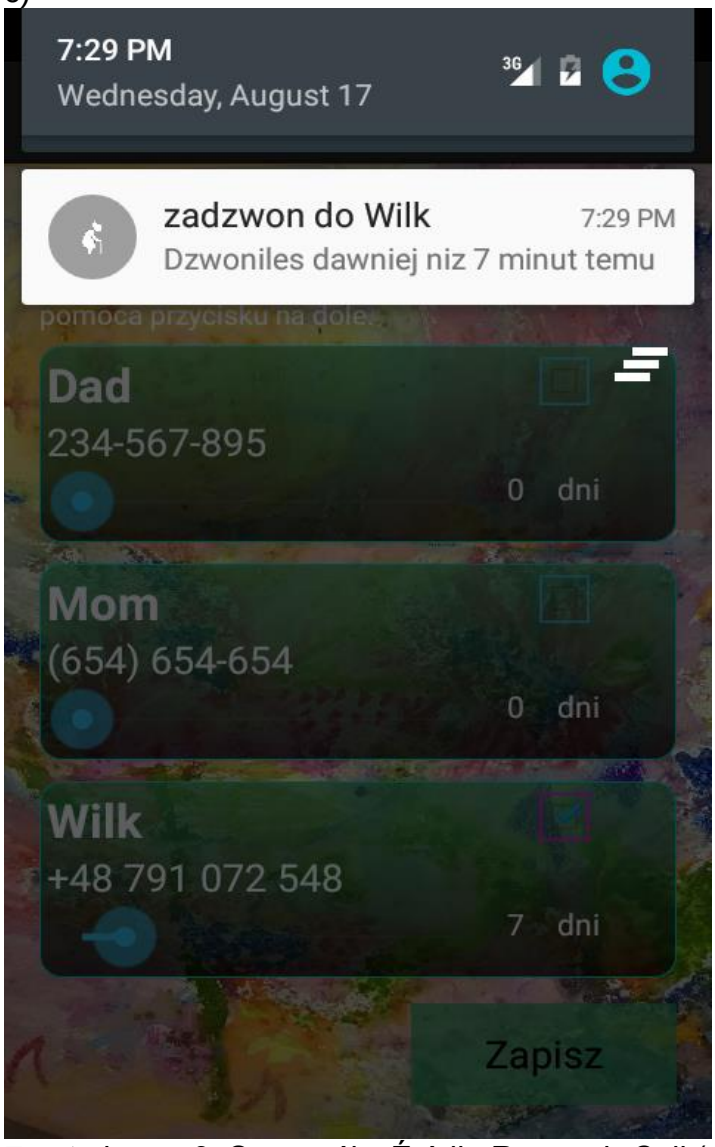
b)



Zrzut ekranu:2: Adnotacja; Źródło:RemembrCall (opracowanie własne)

Przykładowy widok notyfikacji przypominających o wykonaniu telefonu do konkretnej osoby, wysłanych po wybranym przez użytkownika czasie. Adnotacja na samej górze ekranu przybiera kształt spacerującej o lasce staruszki, co nawiązuje tematycznie do ikony całej aplikacji.

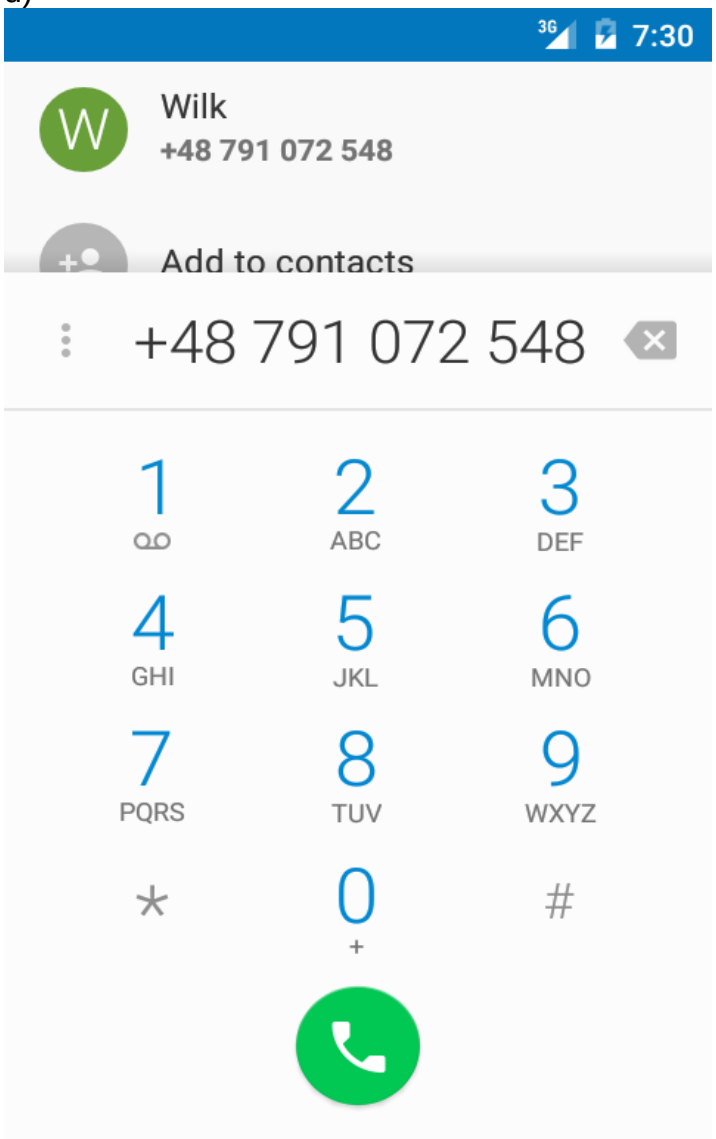
c)



Zrzut ekranu:3: Szczegóły; Źródło:RemembrCall (opracowanie własne)

Po rozwinięciu czarnego paska u góry pokazują się szczegóły poszczególnych notyfikacji takie jak nazwa kontaktu oraz ilość dni (w przykładzie minut), które minęły od ostatniego połączenia z nim.

d)



Zrzut ekranu:4: Wybranie notyfikacji; Źródło:RemembrCall (opracowanie własne)

Po kliknięciu konkretnej adnotacji aplikacja przekierowuje użytkownika bezpośrednio do Książki telefonicznej z już wybranym numerem kontaktu, odnośnie którego notyfikację wybraliśmy. Jest to o tyle komfortowe, że dopóki nie klikniemy adnotacji, będzie nam ona wciąż przypominała o tym, żeby zadzwonić, a gdy zostanie już wybrana poniekąd zmusi użytkownika do wykonania zaplanowanego telefonu.

5.3. Funkcjonalności - fragmenty kodu aplikacji

a) Funkcja *setContactCalls* odpowiedzialna jest za dopasowanie połączeń do konkretnych kontaktów.

```
    while (!phone.isAfterLast()) {

...

        Call recentCall = new Call();
        if(calls != null) {
            for (Call c : calls){
                if(Integer.parseInt(c.getDuration()) > 0){
                    if(c.getDate().compareTo(recentCall.getDate())>0){
                        recentCall = c;
                    }
                }
            }
        }

        if (contact == null) {
            continue;
        }

        contact.setRecentCall(recentCall);

    }
```

b) Funkcja *CallNotification* odpowiedzialna za pokazanie pojedynczej notyfikacji.

```
public class CallNotification {
   ...
   }

   public void showNotification(String message, String tittle, Context context, int id, String phoneNumber)
   ...

       Notification notification = new NotificationCompat.Builder(context)
               .setSmallIcon(R.drawable.babcia)
               .setContentTitle(tittle)
               .setContentText(message)
               .setContentIntent(pi)
               .setAutoCancel(true)
               .build();

       NotificationManager notificationManager = (NotificationManager)
       notificationManager.notify(id, notification);
   }

}
```

c) Główna funkcja odpowiedzialna za wybieranie kontaktów nadających się do wysłania notyfikacji.

```
if(c.isChecked()){
    Log.i("longCall", c.name + " " + c.delay + " od ostatniej rozmowy "+ (c.getProgression()*(1000)));
    String tittle = "zadzwon do " + c.getName();
    String message = "Dzwoniles dawniej niz "+ c.getProgression() + " minut temu";
    String number = "" ;
    if (c.numbers.size() > 0 && c.numbers.get(0) != null) {
        number = c.numbers.get(0).number;
    }

    if(c.getDelay() > c.getProgression()*(1000*60)){
        Log.i("longCall", message);
        new AlarmBroadcaster(context,tittle , message, number, c.id).setAlarmBroadcast(0);
    } else {
        Log.i("longCall", "dzwoniles do " + c.name + " zadzwon za " +

        (c.getProgression()-(c.getDelay()/(1000*60))));
        new AlarmBroadcaster(context, tittle, message,number, c.id)
                .setAlarmBroadcast((int) (c.getProgression()-(c.getDelay()/(1000*60))));
    }

} else {
    new AlarmBroadcaster(context).cancelAlarm(c.id);

}
```

d) Klasa odpowiedzialna za tworzenie alarmów systemowych, które po danym czasie wywołają funkcje wyświetlającą notyfikacje. + możliwość anulowania zakolejkowanych notyfikacji.

```
public void getCallDetails(Context context) {
    ...

        while (callDetailsCursor.moveToNext()) {
        ...


            if(callsMap.get(phNumber) != null)
            {
                ArrayList<Call> calls = callsMap.get(phNumber);
                calls.add(new Call(callDateTime, callDuration, contactName));
                callsMap.put(phNumber, calls);

            } else {
                ArrayList<Call> calls = new ArrayList<>();
                calls.add(new Call(callDateTime, callDuration, contactName));
                callsMap.put(phNumber, calls);
            }

        }
}
```

e) Funkcja tworząca mapę połączeń przypisanych do numeru na podstawie historii połączeń telefonu.

```
   public class AlarmBroadcaster {
   ...

       public AlarmBroadcaster(Context context, String tittle, String message,
           String phoneNumber, String id) {
       ...
           reminderBroadcastIntent = PendingIntent.getBroadcast(context,
           Integer.parseInt(id), intent, PendingIntent.FLAG_UPDATE_CURRENT);

       }

       public void setAlarmBroadcast(int minutes){
       ...
           alarmToBroadcast.set(AlarmManager.RTC_WAKEUP, when, reminderBroadcastIntent);
       }
       public void cancelAlarm(String id){
       ...
           alarmToCancel.cancel(pendingIntentToCancel);
       }
   }
}
```

5.4 Użyte technologie

Aplikacja została napisana w języku programowania Java w środowisku Android Studio w wersji 2.1.. Aplikacja zaprogramowana została z myślą o użytkownikach systemu Android 5.0. bądź późniejszym. Program kompilowany był dla wersji Android API 21., a buildowany dokładnie dla wersji 21.1.2. poprzez gradle 2.1. Odbiór zewnętrznych wiadomości został zrealizowany przez GCM (*ang. Google Cloud Messages*) przy pomocy pushbots w wersji 2.0.13.. Front-end występuje w postaci plików xml (*ang. Extensible Markup Language)* w wersji 1.0. z kodowaniem UTF-8.

Do tworzenia dokumentacji posłużył edytor tekstu *LibreOffice Writer*. Diagram klas stworzono za pomocą programu *Visual Paradigm.*

Jako pomoc przy wspólnym budowaniu projektu wykorzystany został serwis internetowy GitHub z systemem kontroli wersji Git.

Java - obiektowy język programowania służący do tworzenia programów źródłowych stworzony przez James'a Goslinga, pracującego ówcześnie dla *Sun Microsystems.java*.

Android Studio – stworzony przez Google oficjalny IDE (zintegrowane środowisko programistyczne - *ang. Integrated Development Environment)* do tworzenia aplikacji pod system Android.

Android - system operacyjny dla urządzeń mobilnych wprowadzony przez firmę Google.

API - interfejs Programistyczny Aplikacji. Zbiór definicji, protokołów i narzędzi do tworzenia aplikacji.

Gradle - system optymalizacji i automatyzacji procesu kompilacji kodu źródłowego aplikacji na kod maszynowy, również należący do Google.

GCM (*Google Cloud Messaging)* - serwis mobilny (stworzony przez Google) który pozwala przesyłać notyfikacje do użytkowników z zewnętrznego źródła .

PushBots – zaprezentowany przez *PushBots*, Inc darmowy serwis wysyłania notyfikacji typu *push* dla aplikacji mobilnych.

Front-end - warstwa prezentacji, czyli to, co widzi użytkownik aplikacji.

Visual Paradigm - program służący do wizualnego projektowania i modelowania baz danych.

5.5 Testowanie aplikacji

Testy aplikacji były przeprowadzane manualnie, zarówno na emulatorze jak i fizycznych urządzeniach. Pozwoliło to na sprawdzenie działania aplikacji na urządzeniach o różnych specyfikacjach oraz wersjach systemu Android. Testowanie manualne umożliwiło zniwelowanie potrzeby poświęcenia dodatkowego czasu na naukę i opanowanie testów automatycznych, gdyż temat ten był rozwinięty na innej specjalizacji.

Sprawdzenie aplikacji odbywało się przy każdej większej zmianie, pozwalając na wczesne wykrywanie błędów, poprawki i aktualizację kodu. Z myślą o testowaniu RemembrCall na własnych urządzeniach wybraliśmy wersję systemu Android 5.0. Zadbaliśmy więc, by urządzenia fizyczne posiadały odpowiednio wysoką wersję. Tworzenie paczki instalacyjnej oraz instalacja na urządzeniach uproszczone zostały do wciśnięcia jednego przycisku w narzędziu Android Studio.

6. Podział pracy nad projektem

Określenie indywidualnego wkładu w pracę każdego z członków zespołu.

6.1. Tkacz Jessica:

* wczytywanie ustawień dla kontaktów,
* strona wizualna aplikacji (front-end oraz widoki),
* modele kontaktów,
* pobieranie rejestru połączeń,
* sporządzenie dokumentacji

6.2. Wilk Tomasz:

* zapisaniem ustawień dla kontaktów,
* pobieranie kontaktów z urządzenia,
* zarządzaniem notyfikacjami,
* mechanizm opóźniający,
* zarządzanie broadcasterem,
* testowanie aplikacji

7. Podsumowanie

Zgodnie z założeniami stworzyliśmy aplikację prostą w obsłudze, dopasowującą się wymaganiami do osób z różnym stopniem obycia z nowoczesnymi technologiami. Zrealizowane zostały wszystkie funkcjonalności, które wydawały nam się najważniejsze. Na pewno zostały pewne niedociągnięcia, rzeczy które mnożna było zrobić lepiej bądź sprytniej, jednak mimo wszystko jesteśmy zadowoleni z efektów naszej pracy.

Naszym zdaniem największym osiągnięciem była realizacja pomysłu stworzenia mapy połączeń, aby niesamowicie zaoszczędzić czas przeszukiwania bazy kontaktów. Było to zadanie wyjątkowo wymagające, dlatego też pewnie sprawiło nam najwięcej satysfakcji. Zadanie polegało na pobraniu wszystkich kontaktów oraz połączeń, a następnie stworzeniu mapy na zasadzie kontakt->lista połączeń. Po stworzeniu takiej mapy nasz algorytm wybierał najnowsze połączenie i tylko to połączenie zostawało w bazie danych. Dzięki temu, jeżeli użytkownik telefonu wykonałby telefon do jednego z kontaktów, wystarczyło tym połączeniem zastąpić poprzednie połączenie. W efekcie otrzymaliśmy prostą zależność kontakt->najnowsze połączenie, dzięki czemu algorytm nie musi za każdym razem sprawdzać wszystkich połączeń, a odwołuje się bezpośrednio do konkretnego połączenia.

Nie udało nam się natomiast sprawić, by aplikacja działała całkowicie w tle, tzn. po wyrzuceniu jej z ostatnio używanych aplikacji. Wynika to z faktu, iż zbyt późno zorientowaliśmy się, że nie będzie takiej możliwości. Powodem jest wersja systemu Android, dla której pisaliśmy RemembrCall. Okazało się, że trzeba by zmienić dużą część aplikacji by wprowadzić to udogodnienie, co zresztą planujemy zrobić w przyszłości, jako, że udostępniamy naszą pracę na platformie Sklep Play. Nie uważamy tego jednak za zbyt duży problem ze względu na to, że zaobserwowaliśmy, iż znaczna część naszych odbiorców praktycznie w ogóle nie kasuje ostatnio używanych aplikacji.

Praca w dwuosobowej grupie miała swoje plusy i minusy. Dzięki współpracy mogliśmy podzielić między sobą obowiązki, sprawdzić się jako

członek zespołu. Utrudnieniem był fakt, iż oboje pracujemy w bardzo różnych godzinach, praktycznie na zmianę. Ograniczało to ilość spotkań, a nawet rozmów na temat naszej aplikacji. Z czasem udało się nam jednak unormować tę sytuację i bardziej zaangażować w projekt.

Pisanie aplikacji dla systemu Android, a także korzystanie z narzędzi do ich tworzenia było dla nas nowością, czymś z czym nie spotkaliśmy się na studiach. Zależało nam jednak by właśnie dla tego systemu stworzyć aplikację, gdyż oboje jesteśmy jej użytkownikami. Dzięki temu mogliśmy testować RemembrCall na własnych telefonach, a nie tylko na emulatorze w środowisku Android Studio. Mimo początkowych problemów wytrwaliśmy w tym postanowieniu i bardzo się z tego cieszymy. Pomocna była znajomość języka programowania Java, z którym to natomiast mieliśmy wiele do czynienia podczas studiów. Dużym wsparciem dla naszej współpracy okazał się serwis internetowy GitHub, gdyż korzystaliśmy z systemu kontroli wersji Git.

Wyzwanie jakim było pisanie pracy licencjackiej okazało się być bardzo pouczające. Nie tylko ze względu na nowo nabytą wiedzę w zakresie informatyki, ale również jako doświadczenie życiowe. W nowej dla nas sytuacji poznaliśmy inną odsłonę własnej osoby. Dzięki temu na przyszłość w podobnej sytuacji poradzimy sobie sprawniej.

8. Bibliografia

[1] https://en.wikipedia.org/wiki/Front_and_back_ends (19.08.2016)

[2] https://pushbots.org (19.08.2016)

[3] https://pl.wikipedia.org/wiki/Java (19.08.2016)

[4] https://en.wikipedia.org/wiki/Android_(operating_system) (19.08.2016)

[5] https://en.wikipedia.org/wiki/Application_programming_interface (19.08.2016)

[6] https://en.wikipedia.org/wiki/Gradle (13.08.2016)

[7] https://en.wikipedia.org/wiki/Google_Cloud_Messaging (13.08.2016)

[8] https://en.wikipedia.org/wiki/Android_Studio (13.08.2016)

[9] https://pl.wikipedia.org/wiki/Zintegrowane_środowisko_programistyczne (13.08.2016)

[10] http://www.pracedyplomowe.edu.pl/wstep-do-pracy-dyplomowej.html (14.05.2016)

[11] http://www.pracedyplomowe.edu.pl/streszczenie-i-slowa-kluczowe-w-pracy-dyplomowej.html (14.05.2016)

[12] https://github.com/codepath/android-contacts-loader-demo (14.05.2016)

[13] http://stackoverflow.com/questions/7021606/how-to-do-notification-in-android (07.07.2016)

[14] http://stackoverflow.com/questions/10221996/how-do-i-repeat-a-method-every-10- minutes-after-a-button-press-and-end-it-on-ano (07.07.2016)

[15] http://stackoverflow.com/questions/8956218/android-seekbar-setonseekbarchangelistener (07.07.2016)

[16] https://pushbots.com/developer/docs/android-sdk-integration (07.07.2016)

[17] http://stackoverflow.com/questions/12157125/writing-and-reading-string-to-an-internal-storage-in-android (07.07.2016)

[18] https://play.google.com/store/apps/details?id=com.jess.wilu.remembrCall (25.08.2016)

[19] https://www.youtube.com/watch?v=i75aKOg1O3w (19.08.2016)

9. Załączone źródła

- Płyta CD zawierająca kod źródłowy aplikacji, niniejszą dokumentację w wersji elektronicznej oraz krótki film opisujący działanie aplikacji.