

Programowanie urządzeń mobilnych

Projekt:
Mobilny trener

Wykonanie:
Kusz Katarzyna

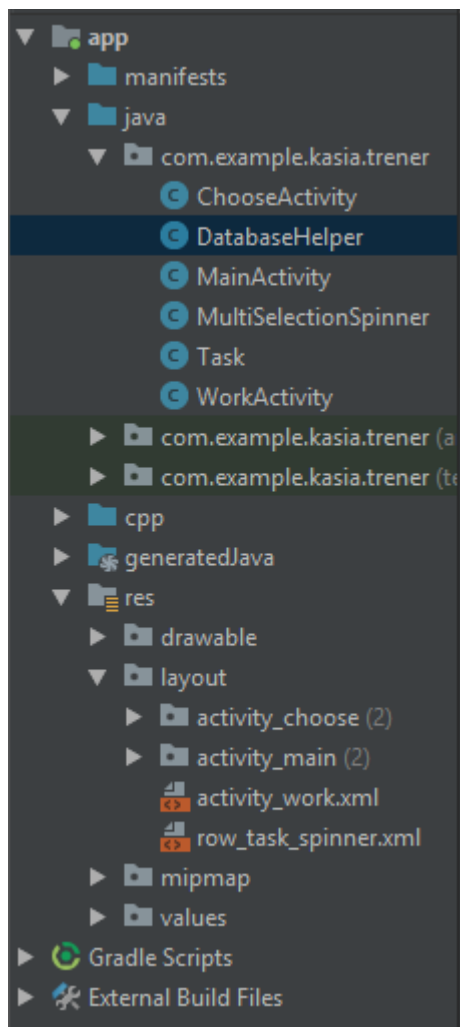
Ogólny zarys aplikacji

Aplikacja została stworzona w programie Android Studio. Na jej potrzeby użyto bazy danych stworzonej lokalnie na urządzeniu przy pomocy SQLite. Całość programu może być używana bez konieczności korzystania z połączenia internetowego.

Stworzenie tej aplikacji miało ułatwić wykonywanie ćwiczeń, które w głównej mierze opierają się na czasowym ich wykonywaniu bez potrzeby liczenia powtórzeń czy serii. Całość miała być prosta w obsłudze i wygodna przy użytkowaniu. Sam zamysł przeszedł kilka modyfikacji i nieco różni się od wersji jaka była planowana na początku, ale końcowo uzyskany został najbardziej praktyczny efekt.

Opis działania aplikacji

Całość aplikacji opiera się na kilku widokach, w których można w prosty sposób dokonać wyborów co do ćwiczeń czasu ich wykonywania lub też dokonywać modyfikacji.



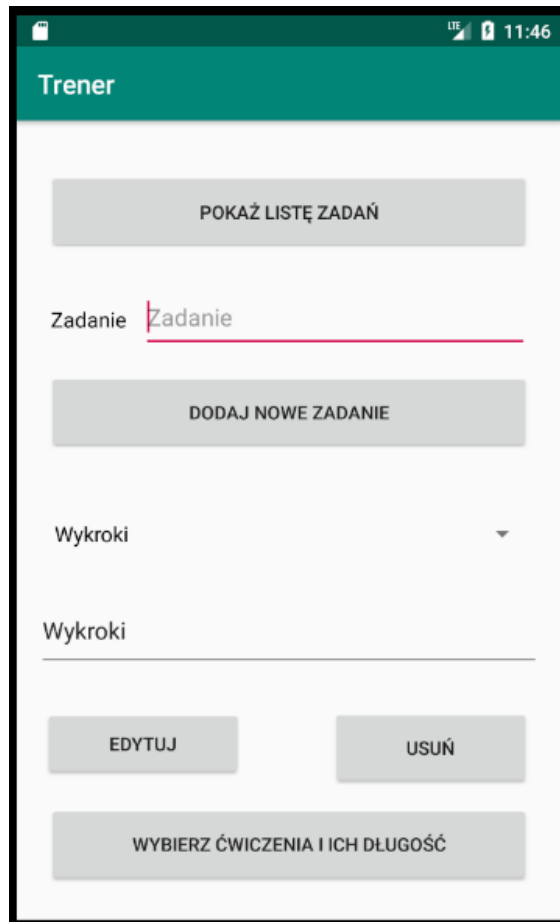
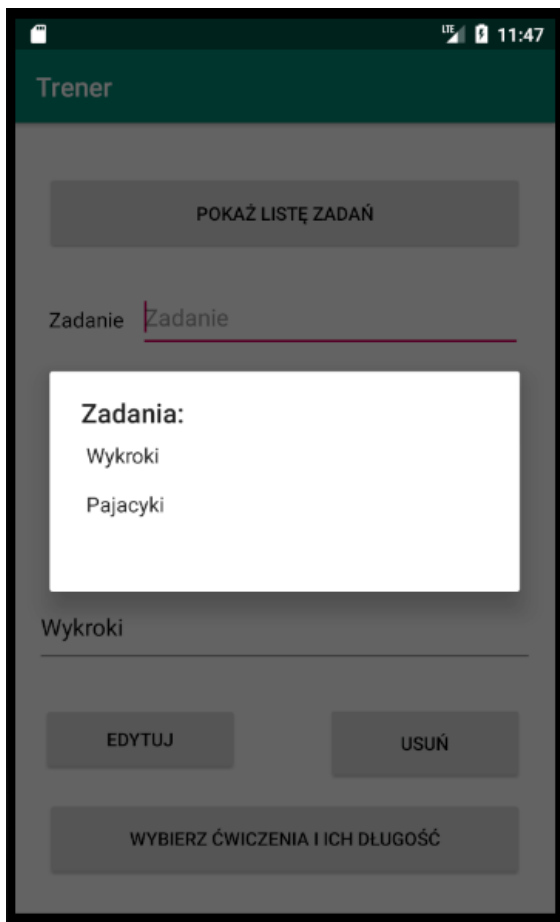
- **MainActivity** – główny widok, który odpala się po uruchomieniu aplikacji

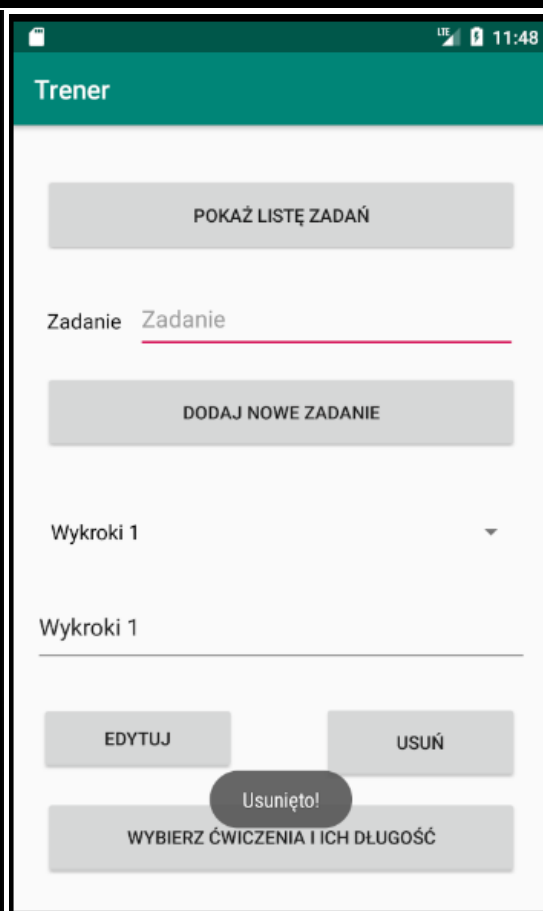
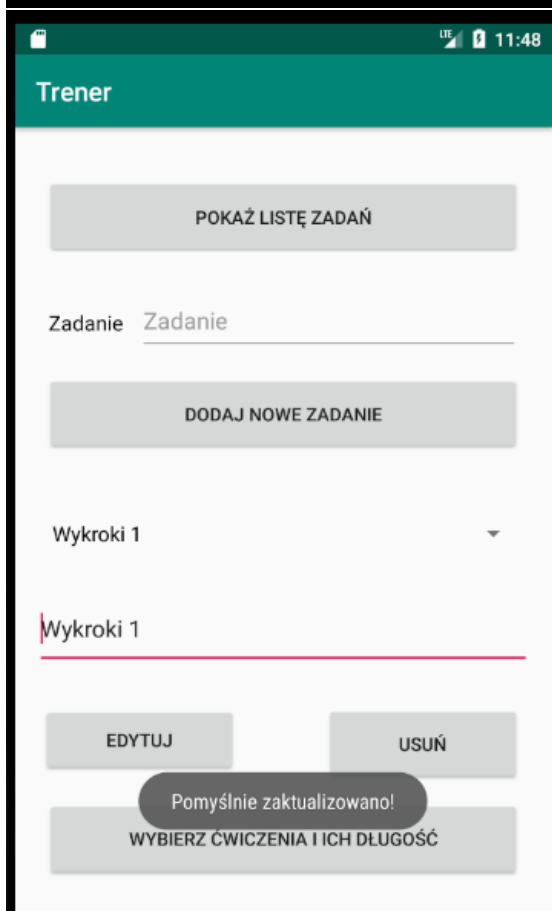
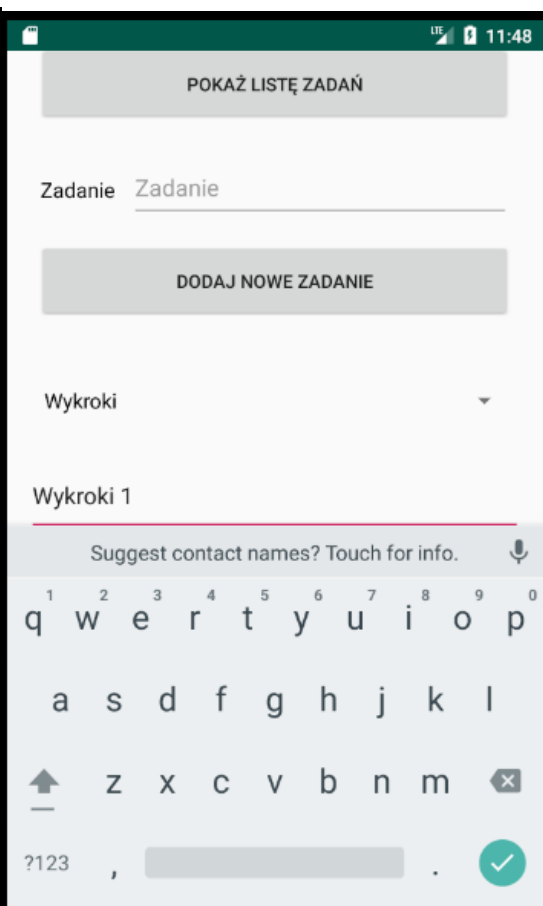
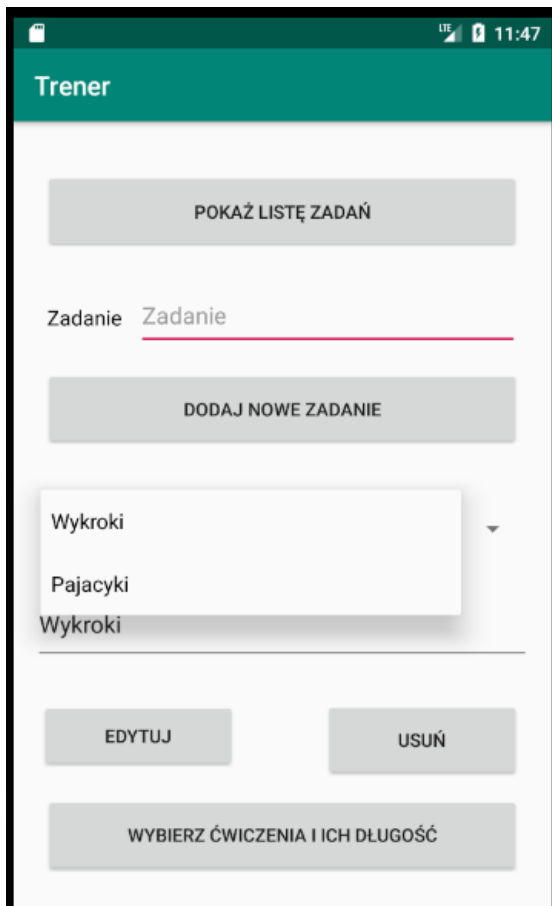
Możliwości jest tu kilka: dodawanie, edytowanie, usuwanie oraz wyświetlanie ćwiczeń.

Dodawanie do bazy jest zabezpieczone przed podaniem pustego pola.

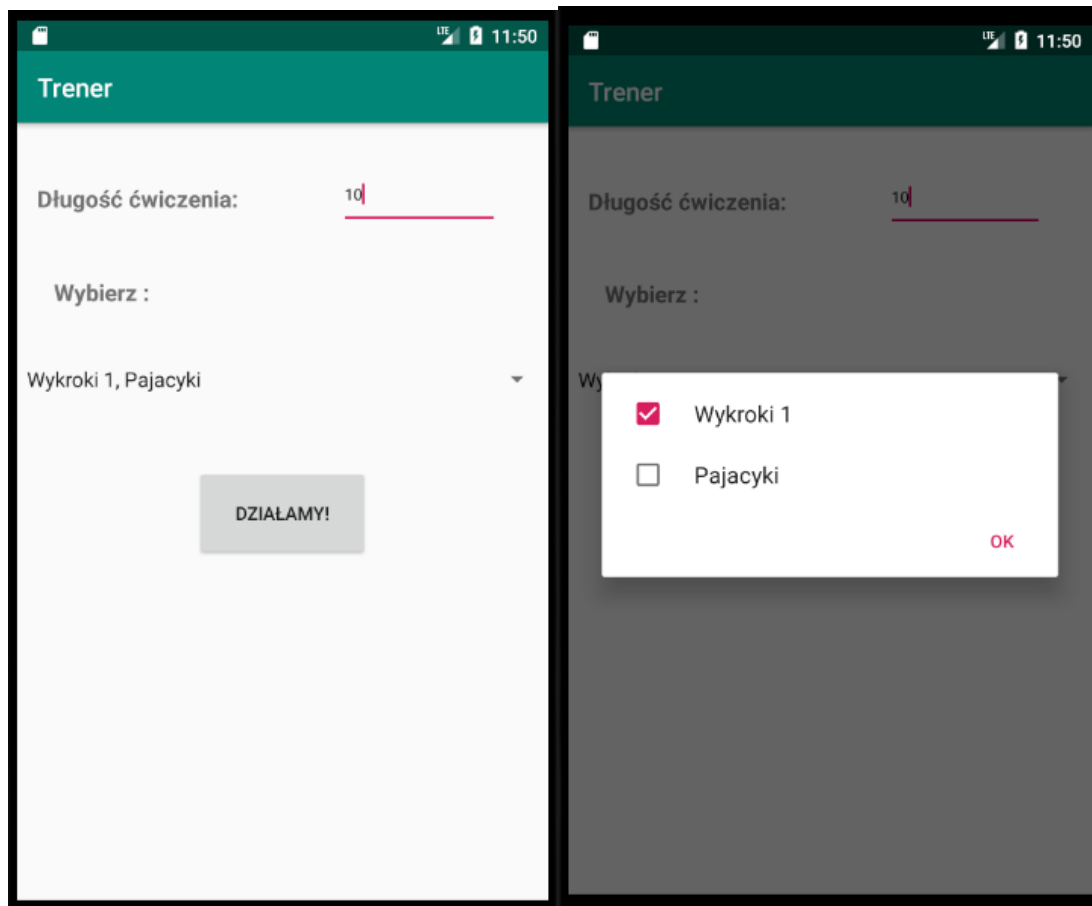
Modyfikowanie jak i usuwanie oparte jest na liście umieszczonej w Spinnerze. Po wybraniu interesującej pozycji podstawiana jest ona w polu tekstowym, gdzie można dokonać edycji lub też usunąć ją z bazy danych aplikacji.

Przycisk „Pokaż listę zadań” działa na zasadzie sczytania wszystkich nazw ćwiczeń z bazy i umieszczeniu ich w wyskakującym oknie tak zwanym Message.





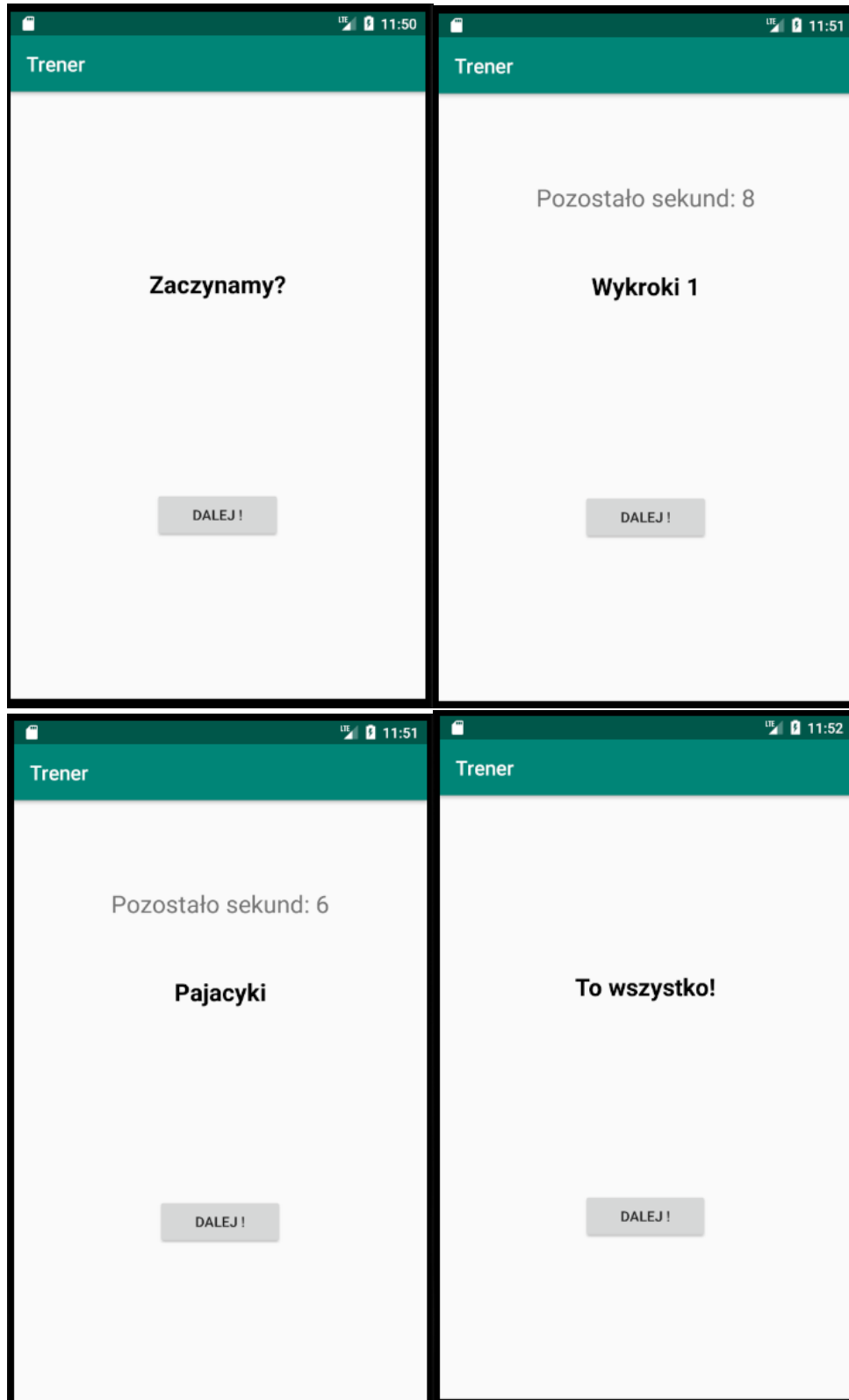
- **ChooseActivity** – w tym widoku podawana jest długość trwania poszczególnego ćwiczenia oraz przy pomocy MultiSpinnera wybierane są z listy ćwiczenia



Pole do podawania długości ćwiczenia jest zabezpieczone przed podaniem niewłaściwych wartości jak litery czy liczby ujemne. Wartości podane są w sekundach.

Dodatkowo sprawdzane jest czy ćwiczenia zostały wybrane z listy, bo w innym wypadku zostaje wyświetlony komunikat błędu.

- **WorkActivity** – w tym oknie wyświetlane są wybrane ćwiczenia przez określony czas, gdy ten czas upłynie następuje odliczanie czasu przerwy i jeśli użytkownik jest gotowy wykonać kolejne ćwiczenie wybiera przycisk „Dalej”. Jeżeli wszystkie zadania z listy zostały wykonane to widoczny będzie napis „To wszystko!”.





Oprócz wyżej wymienionych klas są jeszcze trzy klasy niezbędne do działania aplikacji.




- **DatabaseHelper** – są tu wszelkie metody używane przy operacji na bazie danych czyli uzyskiwanie danych, aktualizowanie, usuwanie i inne.
- **Task** – w tej klasie są tak zwane settery i gettery do konkretnych kolumn w tabeli w bazie danych
- **MultiSelectionSpinner** – to klasa niezbędna do utworzenia Spinnera z wielokrotnym wyborem w widoku ChooseActivity. Są w niej metody pozwalające na wyświetlenie zadań na liście oraz następnym ich wyborze.


Testy interfejsu aplikacji


Aplikacja została poddana testom, które oferuje framework Firebase. Test Lab jest narzędziem w pełni wspierającym platformę Androida, który w prosty sposób umożliwia przetestowanie aplikacji pod kątem działania interfejsu. Testy mogą być napisane własnoręcznie przez programistę, ale także istnieje możliwość użycia testów automatycznych tzw. „test Robo”.


Chcesz wykonywać nieograniczoną liczbę testów? [Zmień plan rozliczeniowy](#)


Trener

Zestaw testów	Typ testu	Rozpoczęto	Łączna liczba wykonań
 matrix-xpl07vw2ktqwa	Robo	4 godz. temu	1
 matrix-2rc6bzl3pnemk	Robo	4 godz. temu	2
 matrix-1f4tqytgaenn3	Robo	4 godz. temu	1

 Uruchom test Robo

 Przeprowadź test narzędzi

 Uruchom pętlę gry

 Przeprowadź XCTest

Pomóż mi wybrać

Dzięki temu, że Firebase jest narzędziem od platformy Google, to testowanie można przeprowadzić na urządzeniach fizycznych jak i emulatorach udostępnionych właśnie przez Google. Ciekawszym wyborem jest wybór urządzenia fizycznego, ponieważ w następstwie tego wyniki są bardziej rozbudowane i dodatkowo przeprowadzone zostaje kilka analiz naszej aplikacji w oparciu o działanie jej na urządzeniu.

Konfiguracja testu

2 Wybierz wymiary

Wybierz urządzenia, poziomy interfejsu API, orientacje oraz ustawienia regionalne, w przypadku których chcesz przeprowadzić test. Musisz wybrać przynajmniej jeden wymiar każdego rodzaju.

Dzienny limit uruchomień testów na urządzeniach fizycznych to 5, a na wirtualnych 10 [Uaktualnij](#)

Urządzenia

Urządzenia fizyczne

Telefony

Essential PH-1 Essential Products

☐ 25

Pixel Google

☐ 25 ☐ 26 ☐ 27 ☐ 28

Pixel 2 Google

☐ 26 ☐ 27 ☒ 28

Pixel 2 XL Google

☐ 26 ☐ 27

Urządzenia wirtualne

Telefony

Low-resolution MDPI phone Generic

☐ 23 ☐ 24 ☐ 25 ☐ 26 ☐ 27 ☐ 28

Pixel 2 Google

☐ 26 ☐ 27 ☐ 28

Nexus 6P Google

☐ 23 ☐ 24 ☐ 25 ☐ 26 ☐ 27

Nexus 4 LG

☐ 19 ☐ 21 ☐ 22

Orientacja

- ☐ Poziomo
- ☒ Pionowo

Ustawienia regionalne

polski [pl] X

Dodaj ustawienie regionalne...

Opcje zaawansowane



Czas oczekiwania testu

Maksymalny czas działania testu (w minutach).

5



Dane logowania konta testowego (opcjonalne)

Jeśli Twoja aplikacja wymaga niestandardowego logowania, wpisz nazwy zasobów elementów logowania oraz dane logowania.

Wpisz nazwę pola, które zaw

Wpisz nazwę użytkownika

Wpisz nazwę pola, które zaw

Wpisz hasło

Dodatkowe pola (opcjonalne)

Jeśli Twoja aplikacja zawiera dodatkowe elementy, które wymagają tekstu wejściowego, wpisz poniżej nazwy zasobów i wejściowe ciągi znaków.

Wpisz nazwę pola

Wpisz wartość



Dodaj pole

Precyzyjne linki (opcjonalnie)

Wpisane poniżej adresy URL zostaną przetestowane przez robota Robo.

Wpisz URL



Dodaj precyzyjny link

☐ Zapisz szablon 1 urządzenie . 1 orientacja . 1 ustawienie regionalne

Konfiguracja i pierwsze uruchomienie testu

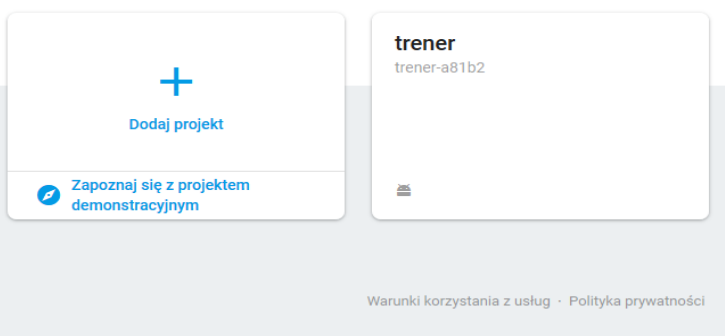
1. Pierwszym krokiem by skonfigurować aplikację jest wejście na stronę Firebase i utworzenie nowego projektu.

Witamy w Firebase

Narzędzia Google do tworzenia doskonałych aplikacji,
kontaktowania się z użytkownikami i zwiększania przychodów
z reklam mobilnych.

[Więcej informacji](#) [Dokumentacja](#) [Pomoc](#)

Najnowsze projekty



2. Należy podać nową nazwę dla projektu i zaakceptować warunki.

The screenshot shows the 'Dodawanie projektu' (Adding project) dialog in the Firebase console. The dialog has a title bar with a close button. It contains the following fields and options:

- Nazwa projektu**: A dropdown menu with the value 'nazwaProjektu'.
- Identyfikator projektu**: A text field with the value 'nazwaprojektu-b7120' and a pencil icon for editing.
- Lokalizacja**: A text field with the value 'nam5 (us-central)' and a pencil icon for editing. The text '(Analytics)' and '(Cloud Firestore)' are shown next to it.
- Udostępniaj dane Google Analytics dla Firebase przy użyciu domyślnych ustawień**: A checked checkbox.
- Akceptuję**: A checked checkbox.
- Warunki współpracy w zakresie ochrony danych między Google a właścicielami kont usług pomiarowych występującymi w roli administratorów**: A link to the terms of service.
- Więcej informacji**: A link to more information.

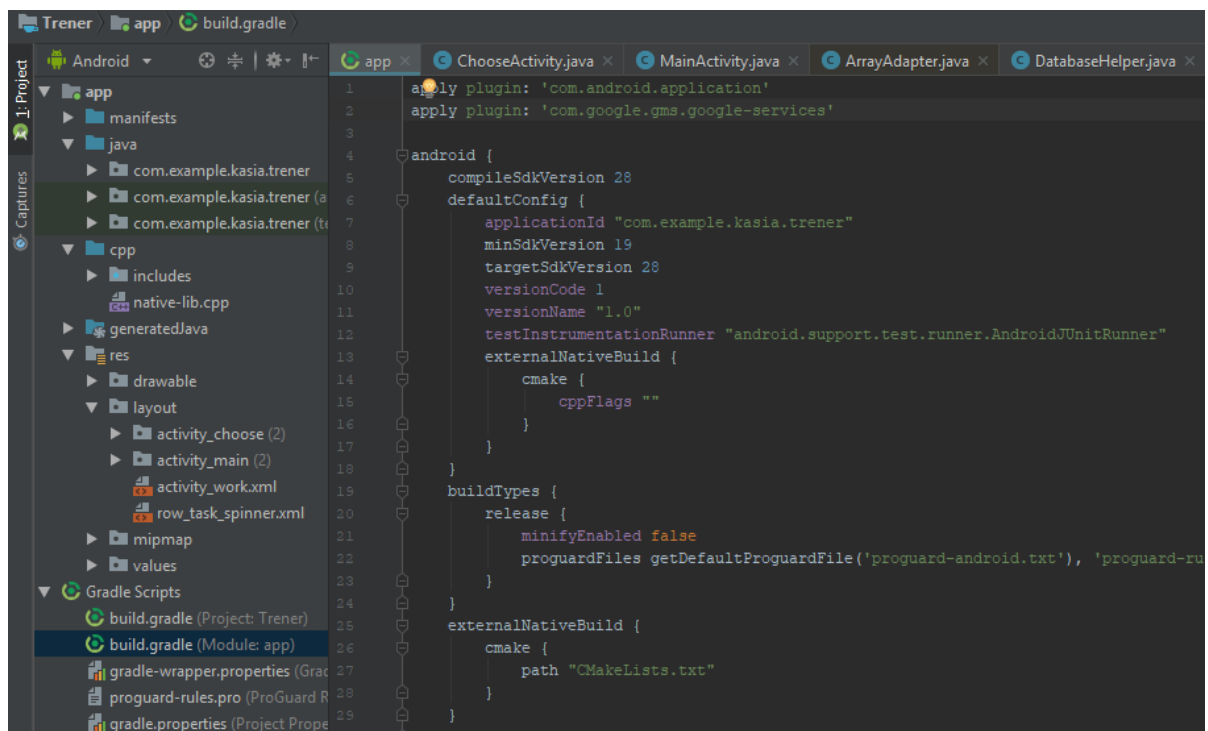
At the bottom of the dialog, there are two buttons: 'Anuluj' (Cancel) and 'Utwórz projekt' (Create project).

3. Ze strony Firebase niezbędny do konfiguracji jest plik google-services.json, dlatego trzeba go pobrać i umieścić w odpowiednim folderze aplikacji.

Ten komputer > Dysk (E:) > Studia > Trener_beta > Trener > app

Nazwa	Data modyfikacji	Typ	Rozmiar
.externalNativeBuild	24.01.2019 18:48	Folder plików	
build	24.01.2019 18:48	Folder plików	
libs	02.12.2018 14:08	Folder plików	
src	24.01.2019 18:48	Folder plików	
.gitignore	02.12.2018 14:08	Dokument tekstowy	1 KB
app.iml	25.01.2019 21:13	Plik IML	19 KB
build.gradle	25.01.2019 21:12	Plik GRADLE	2 KB
CMakeLists.txt	02.12.2018 14:10	Dokument tekstowy	2 KB
google-services.json	25.01.2019 13:49	JSON File	1 KB
proguard-rules.pro	02.12.2018 14:08	Prolog Source	1 KB

4. Aplikacja, która poddawana jest testom należy do platformy Android i tak należy wybrać. Konieczne jest podanie nazwy pakietu aplikacji, którą można znaleźć w build.gradle jako applicationId.




```
1  apply plugin: 'com.android.application'
2  apply plugin: 'com.google.gms.google-services'
3
4  android {
5      compileSdkVersion 28
6      defaultConfig {
7          applicationId "com.example.kasia.trener"
8          minSdkVersion 19
9          targetSdkVersion 28
10         versionCode 1
11         versionName "1.0"
12         testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
13         externalNativeBuild {
14             cmake {
15                 cppFlags ""
16             }
17         }
18     }
19     buildTypes {
20         release {
21             minifyEnabled false
22             proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
23         }
24     }
25     externalNativeBuild {
26         cmake {
27             path "CMakeLists.txt"
28         }
29     }
30 }
```

Dodaj Firebase do swojej aplikacji dla systemu Android

123

Register appDownload config fileAdd Firebase SDK


Rozpocznij szybciej na Androidzie, klikając [Narzędzia > Firebase](#) w [Android Studio 2.2](#) lub [jego nowszej wersji](#)

Nazwa pakietu ⓘ

com.twojaaplikacja.android

Pseudonim aplikacji (opcjonalny) ⓘ

Aplikacja freemium na Androida

Certyfikat SHA-1 do podpisania aplikacji przed debugowaniem (opcjonalnie) ⓘ

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00

Wymagane do obsługi Dynamic Links, Invites i Google Sign-In w Auth. Edytuj SHA-1 w ustawieniach.

ANULUJ

DODAJ APLIKACJĘ

pobiera
google-services.json dla

5. W tym samym pliku dopisać trzeba linijkę 2 oraz 40.

```

31
32 dependencies {
33     implementation fileTree(dir: 'libs', include: ['*.jar'])
34     implementation 'com.android.support:appcompat-v7:28.0.0'
35     implementation 'com.android.support.constraint:constraint-layout:1.1.3'
36     testImplementation 'junit:junit:4.12'
37     androidTestImplementation 'com.android.support.test:runner:1.0.2'
38     androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
39     implementation 'com.android.support:recyclerview-v7:28.0.0'
40     implementation 'com.google.firebase:firebase-core:16.0.1'
41 }
42

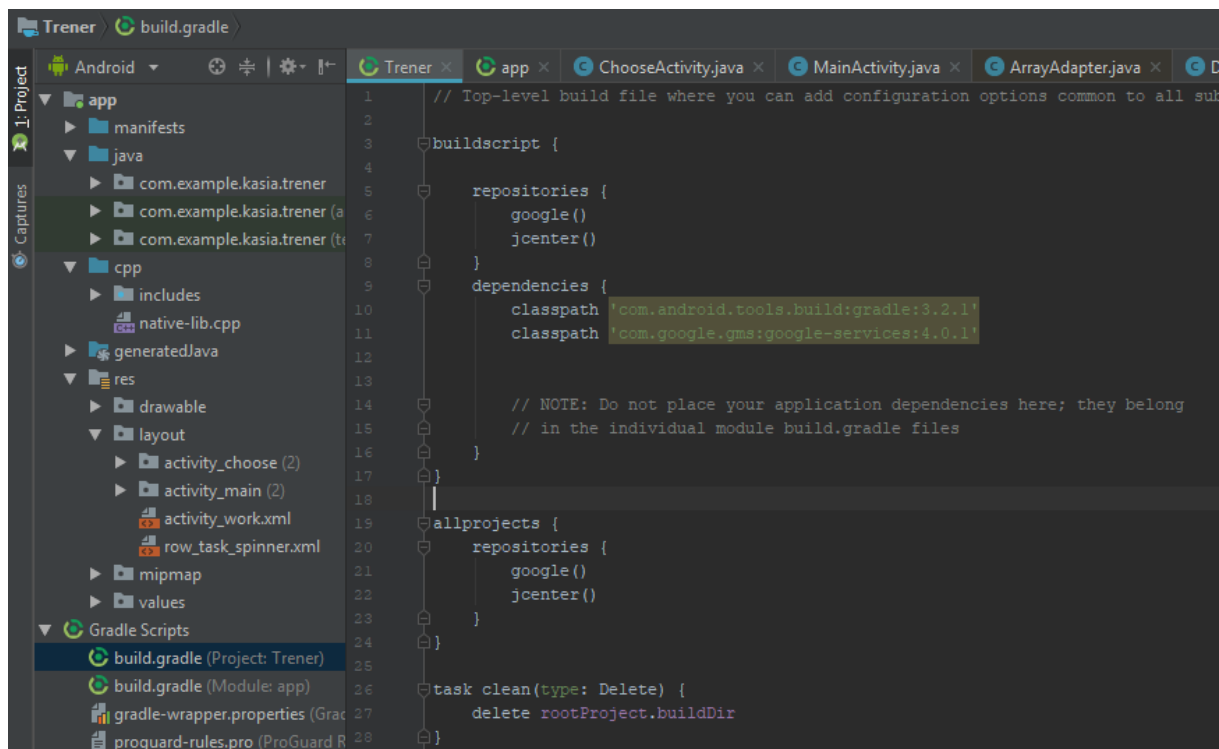
```

```

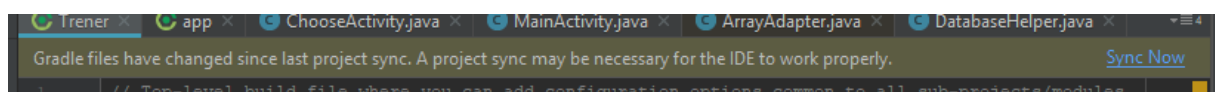
1 apply plugin: 'com.android.application'
2 apply plugin: 'com.google.gms.google-services'
3
4 android {
5     compileSdkVersion 28
6     defaultConfig {
7         applicationId "com.example.kasia.trener"
8         minSdkVersion 19

```

Do pliku build.gradle na poziomie aplikacji dodać należy także linijkę 11.

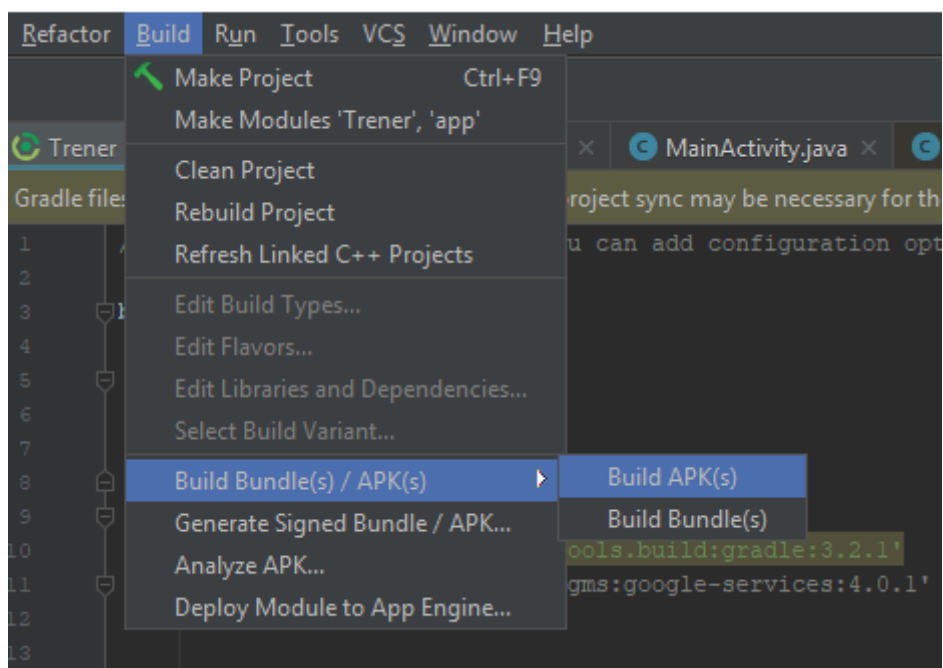


Gdy zostaną dodane powyższe linijki kodu oraz plik json zostanie przeniesiony do folderu aplikacji można przystąpić do synchronizacji projektu, co będzie łatwe dzięki pojawieniu się komunikatu zachęającego do tego. Wystarczy nacisnąć „Sync Now”.



6. Z tak przygotowanej aplikacji konieczne jest teraz uzyskanie pliku apk, który następnie będzie wykorzystany przy testach. W Prawym dolnym rogu programu pojawi się komunikat o utworzeniu pliku i możliwości przejścia do jego lokalizacji.

er - Android Studio



10. Po ukończeniu testu dostępne są różne zakładki z informacjami uzyskanymi podczas testowania.

Test Lab > Trener > matrix-11u7w8bb0fnz6

← Robo test, Nexus 5, Virtual, poziom API 21

✓ Zaliczone

25.01.2019, 22:07

3 min 12 s

Pionowo

polski

Robo

Logi

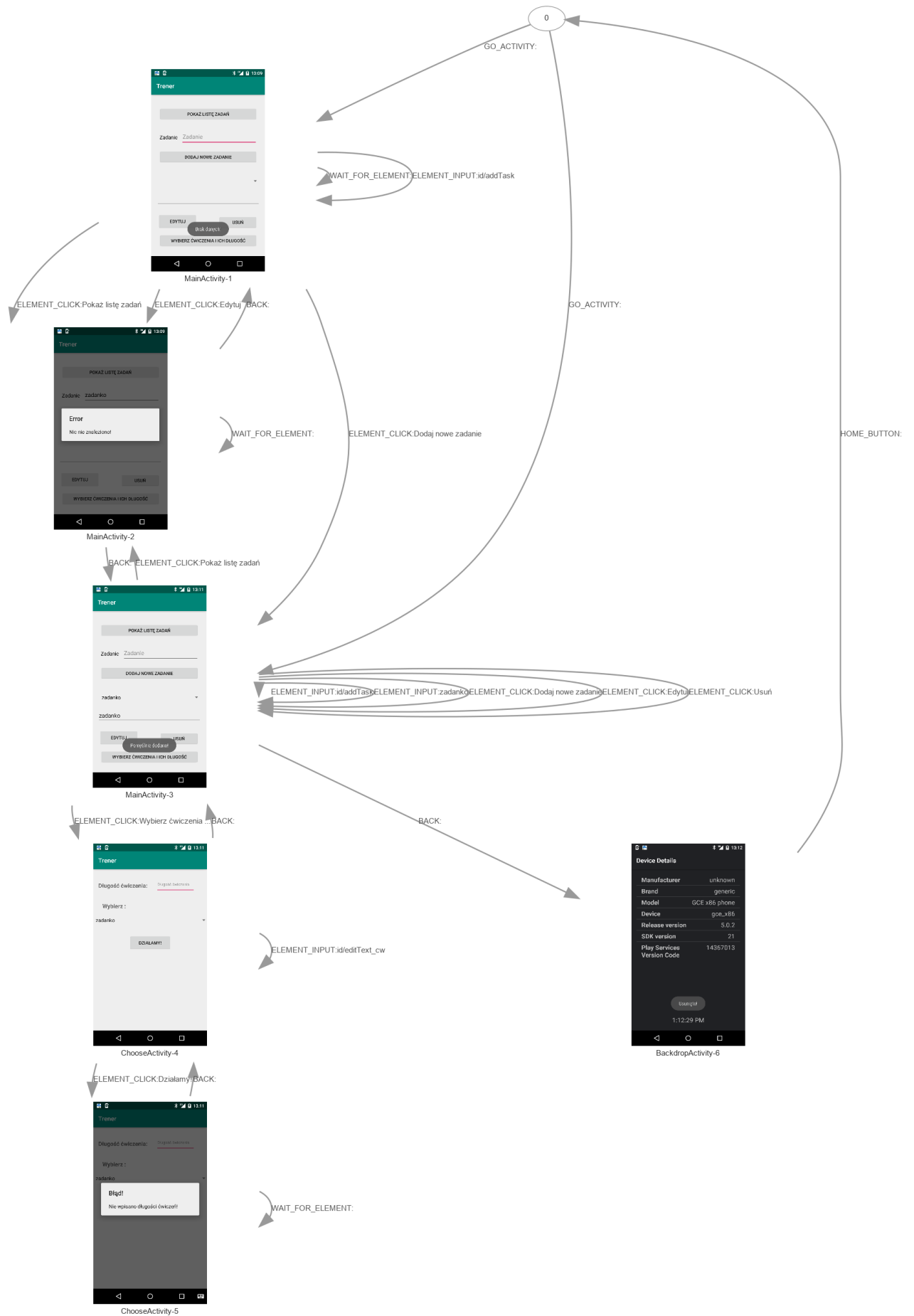
Zrzuty ekranu

Filmy

Wyniki

Czas trwania indeksowania	Statystyki indeksowania ?		
3 min 5 s	Czynności 29	Działania 2	Ekrany 6

Poniżej zamieszczona jest mapka jaka powstała podczas testowania przez robota interfejsu aplikacji. Jak widać nie wszystkie możliwości aplikacji zostały wykorzystane, ale dzięki temu, że te działania są losowe i każdy test wygląda inaczej, to zostają wychwycone błędy jakie mogły zostać pominięte podczas pisania programu.



Podsumowanie

Samo pisanie aplikacji było dość czasochłonne, bo jednak czasem to co się planuje nie zawsze da się wykonać właśnie w taki sposób i nie raz trzeba zmienić lub nagiąć koncepcję. Wiele czynników należy brać pod uwagę jak i sposób w jaki aplikacja działa, tak aby ograniczyć ryzyko występowania błędów i końcowo je wyeliminować. Podczas finalnych poprawek aplikacji jeszcze przed testami aplikacja uruchamiała się zarówno w emulatorze jak i na fizycznym telefonie. Dopiero uruchomienie testów pokazało, że pewne metody były nie do końca poprawnie napisane i tworzyły ryzyko wystąpienia błędu choć jak widać ciężkiego do wykrycia bez testowania interfejsu w tym wypadku przez automatycznego robota.