Programowanie urządzeń mobilnych

Projekt:
Mobilny trener

Wykonanie:
Kusz Katarzyna

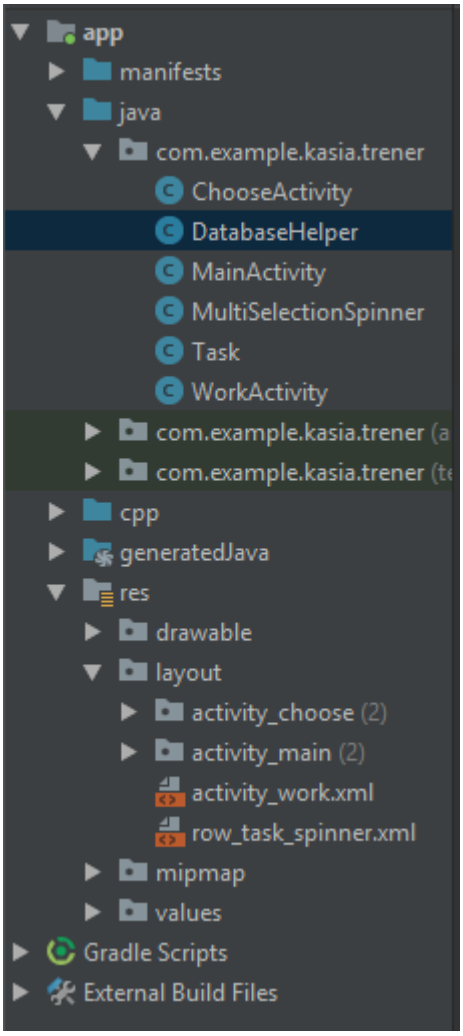
Ogólny zarys aplikacji

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.



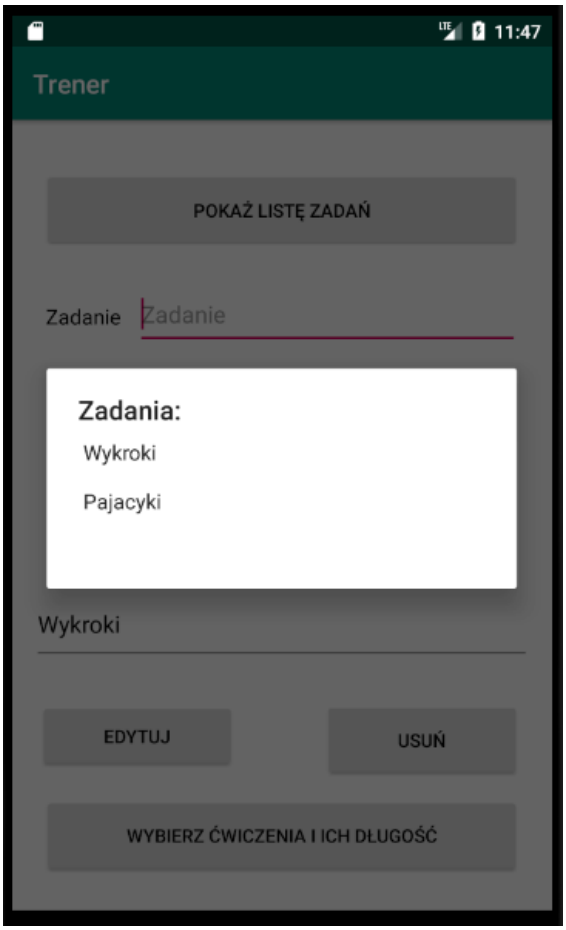
- **MainActivity** – główny widok, który odpala się po uruchomieniu aplikacji

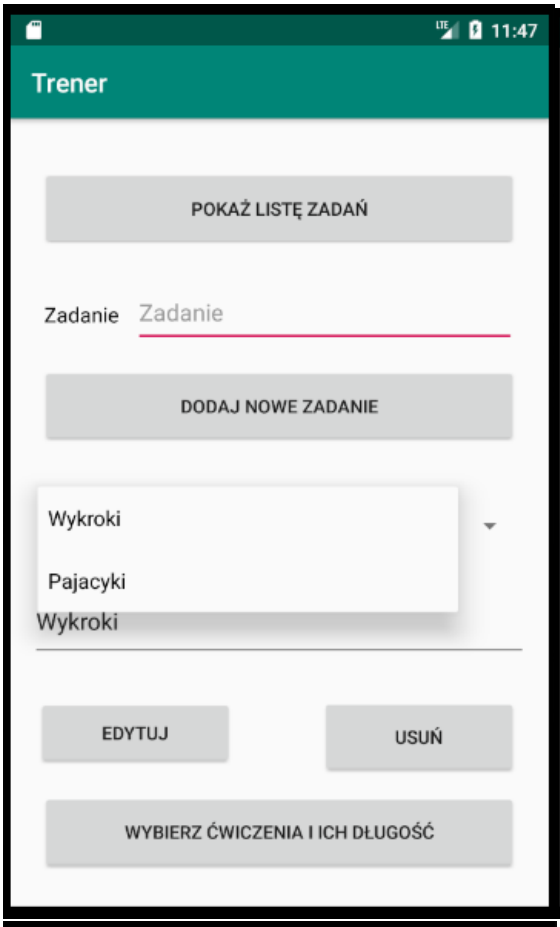
Możliwości jest tu kilka: dodawanie, edytowanie, usuwanie oraz wyświetlanie ćwiczeń.

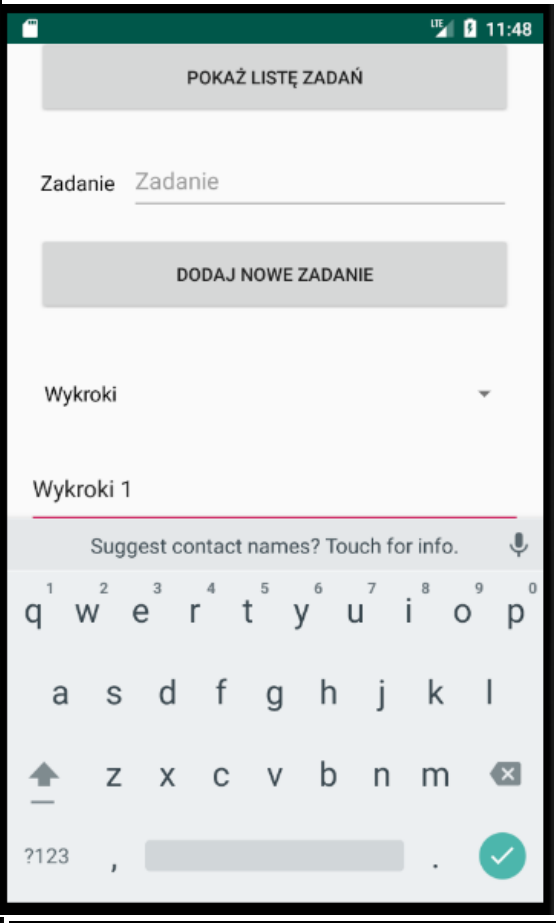
Dodawanie do bazy jest zabezpieczone przed podaniem pustego pola.

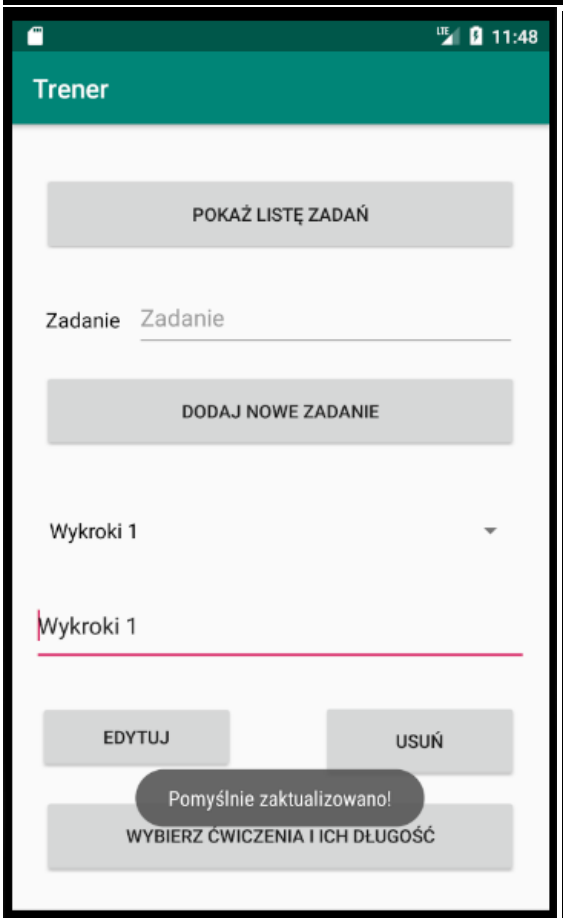
Modyfikowanie jak i usuwanie oparte jest na liście umieszczonej w Spinnerze. Po wybraniu interesującej pozycji podstawiana jest ona w polu tekstowym, gdzie można dokonać edycji lub też usunąć ją z bazy danych aplikacji.

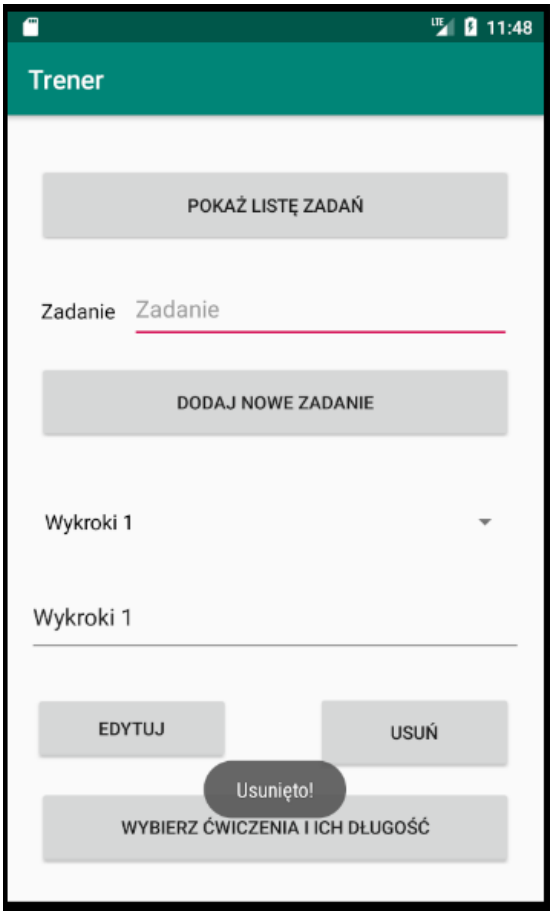
Przycisk „Pokaż listę zadań” działa na zasadzie sczytania wszystkich nazw ćwiczeń z bazy i umieszczeniu ich w wyskakującym oknie tak zwanym Message.



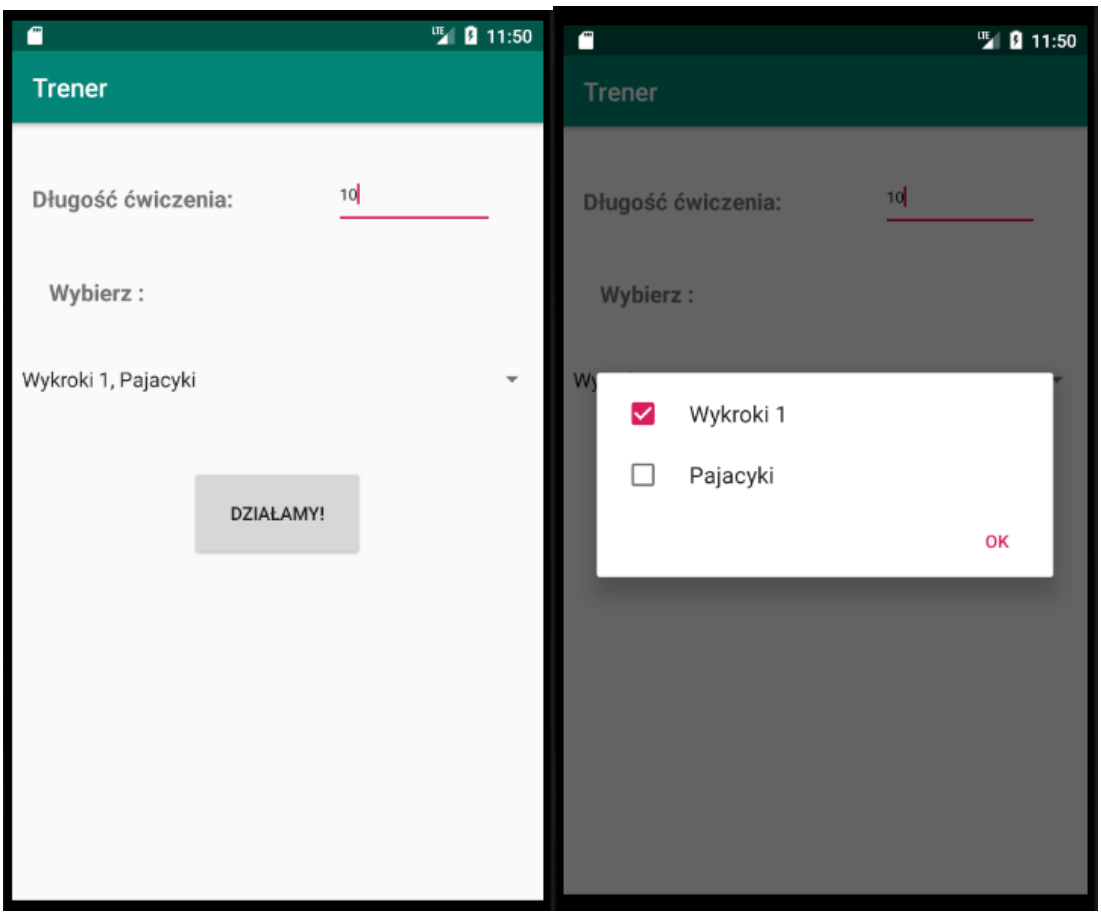

11:47
Trener
POKAŻ LISTĘ ZADAŃ
Zadanie
Zadanie
DODAJ NOWE ZADANIE
Wykroki
Pajacyki
Wykroki
EDYTUJ
USUŃ
WYBIERZ ĆWICZENIA I ICH DŁUGOŚĆ


11:48
POKAŻ LISTĘ ZADAŃ
Zadanie
Zadanie
DODAJ NOWE ZADANIE
Wykroki
Wykroki 1
Suggest contact names? Touch for info.
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123


11:48
Trener
POKAŻ LISTĘ ZADAŃ
Zadanie
Zadanie
DODAJ NOWE ZADANIE
Wykroki 1
Wykroki 1
EDYTUJ
USUŃ
Pomyślnie zaktualizowano!
WYBIERZ ĆWICZENIA I ICH DŁUGOŚĆ


11:48
Trener
POKAŻ LISTĘ ZADAŃ
Zadanie
Zadanie
DODAJ NOWE ZADANIE
Wykroki 1
Wykroki 1
EDYTUJ
USUŃ
Usunięto!
WYBIERZ ĆWICZENIA I ICH DŁUGOŚĆ

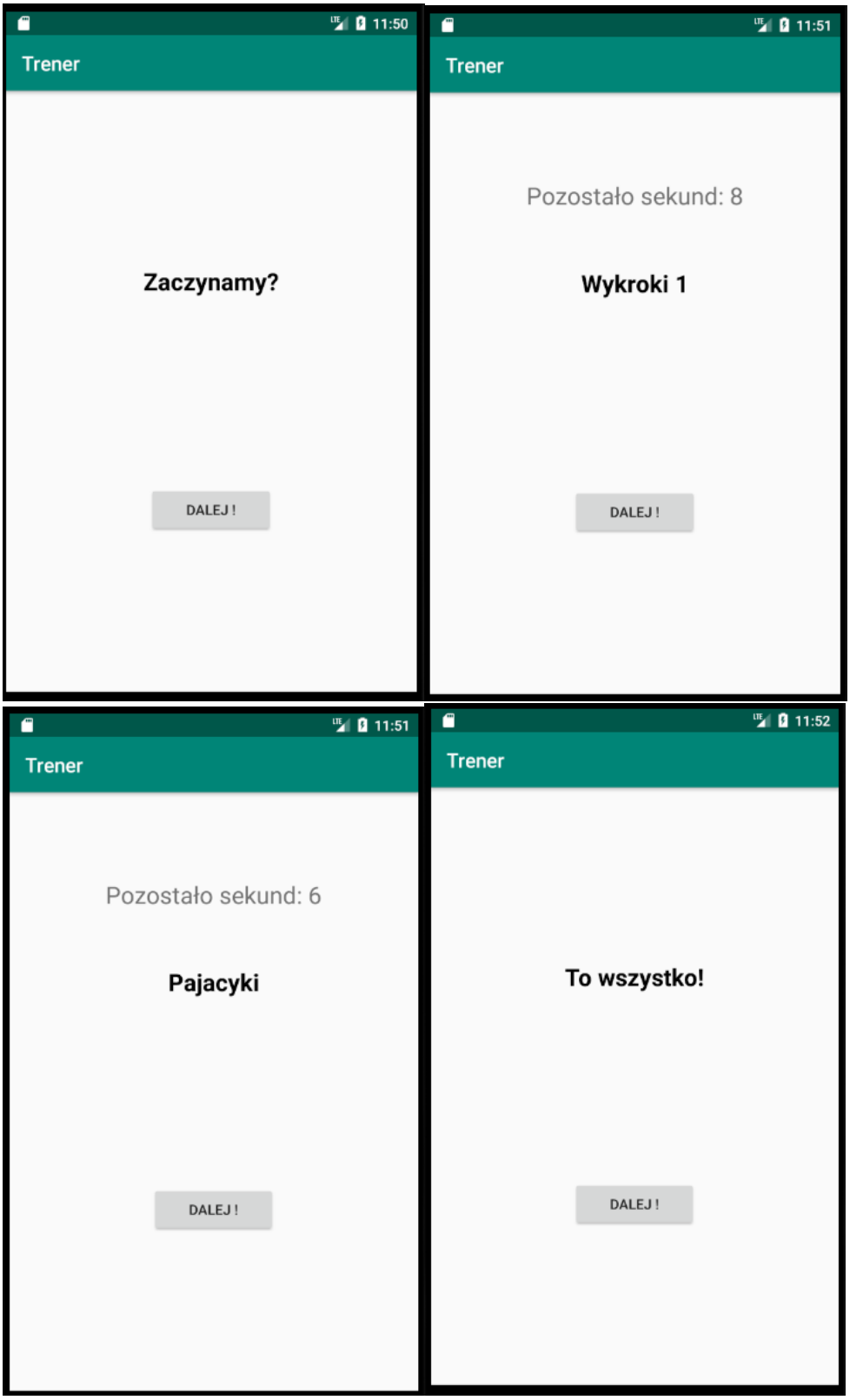
- **ChooseActivity** – w tym widoku podawana jest długość trwania poszczególnego ćwiczenia oraz przy pomocy MultiSpinnera wybierane są z listy ćwiczenia



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.

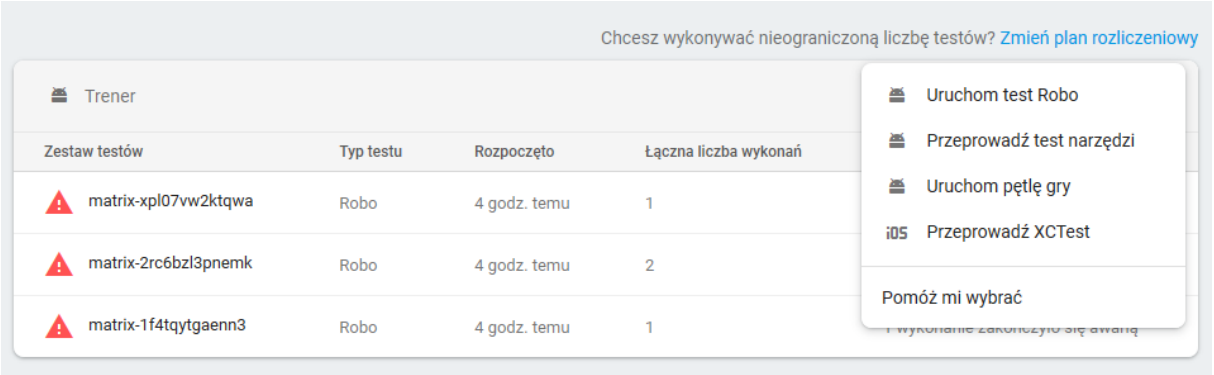
- **DatabaseHelper** – są tu wszelkie metody używane przy operacji na bazie danych czyli uzyskiwanie danych, aktualizowanie, usuwanie i inne.

- **Task** – w tej klasie są tak zwane settery i gettery do konkretnych kolumn w tabeli w bazie danych

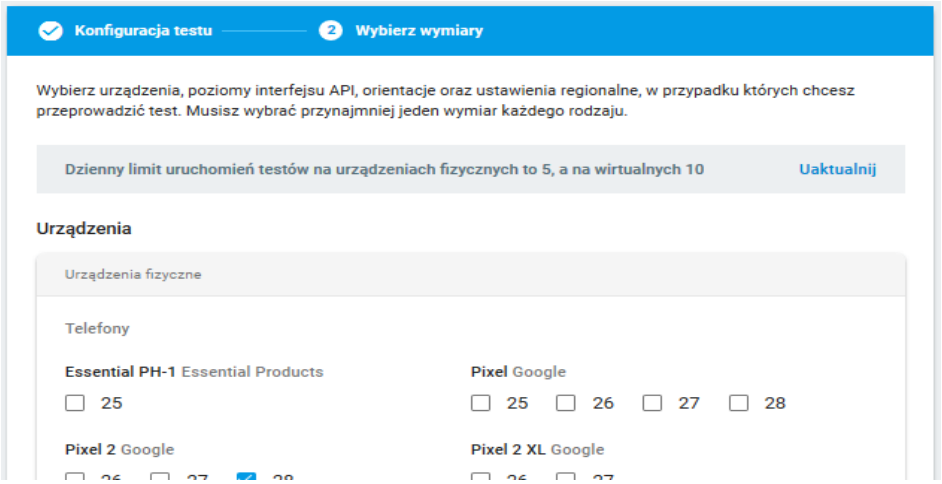
- **MultiSelectionSpinner** – to klasa niezbędna do utworzenia Spinnera z wielokrotnym wyborem w widoku ChooseActivity. Są w niej metody pozwalające na wyświetlenie zadań na liście oraz następnym ich wyborze.

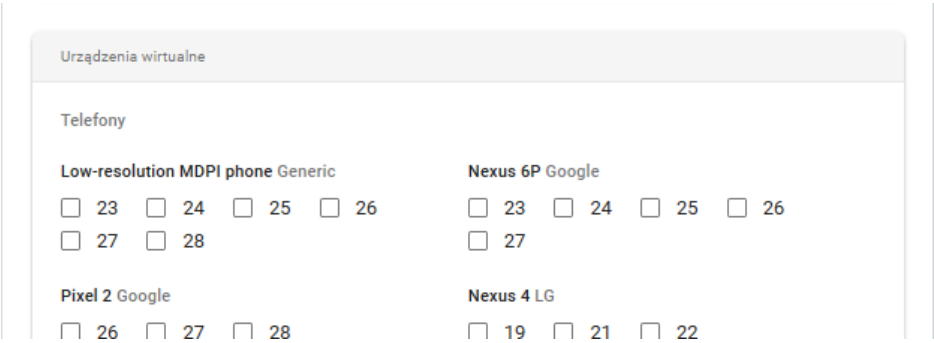
Testy interfejsu aplikacji

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”.



Dzięki temu, że Firebase jest narzędziem od platformy Google, to testowanie można przeprowadzić na urządzeniach fizycznych jak i emulatorach udostępnionym 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.





Orientacja

- [ ] Poziomo
- [x] Pionowo

Ustawienia regionalne

polski [pl] ✕ Dodaj ustawienie regional...

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.

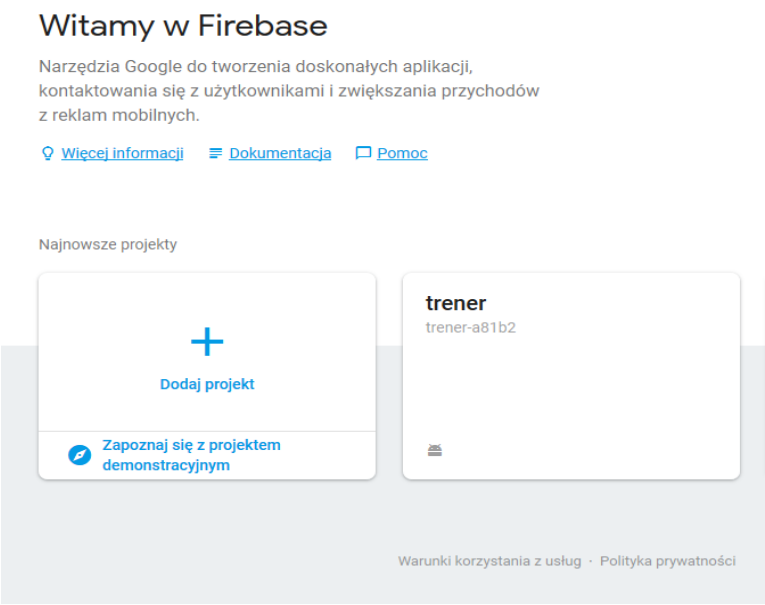
Wpisz URL

Dodaj precyzyjny link

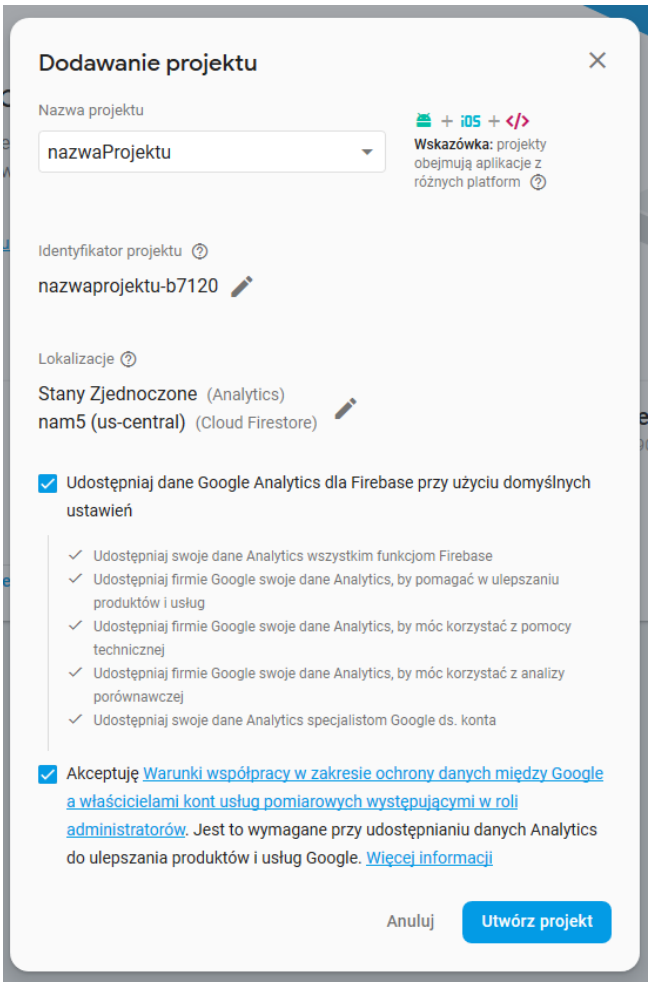
Zapisz szablon 1 urządzenie , 1 orientacja , 1 ustawienie regionalne

Konfiguracja i pierwsze uruchomienie testu

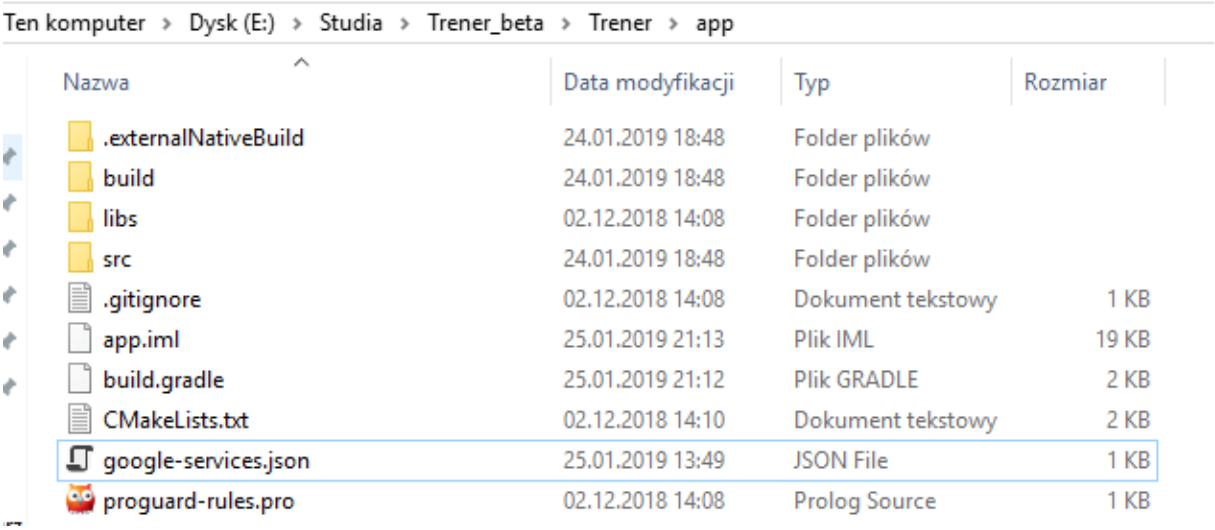
1. Pierwszym krokiem by skonfigurować aplikację jest wejście na stronę Firebase i utworzenie nowego projektu.



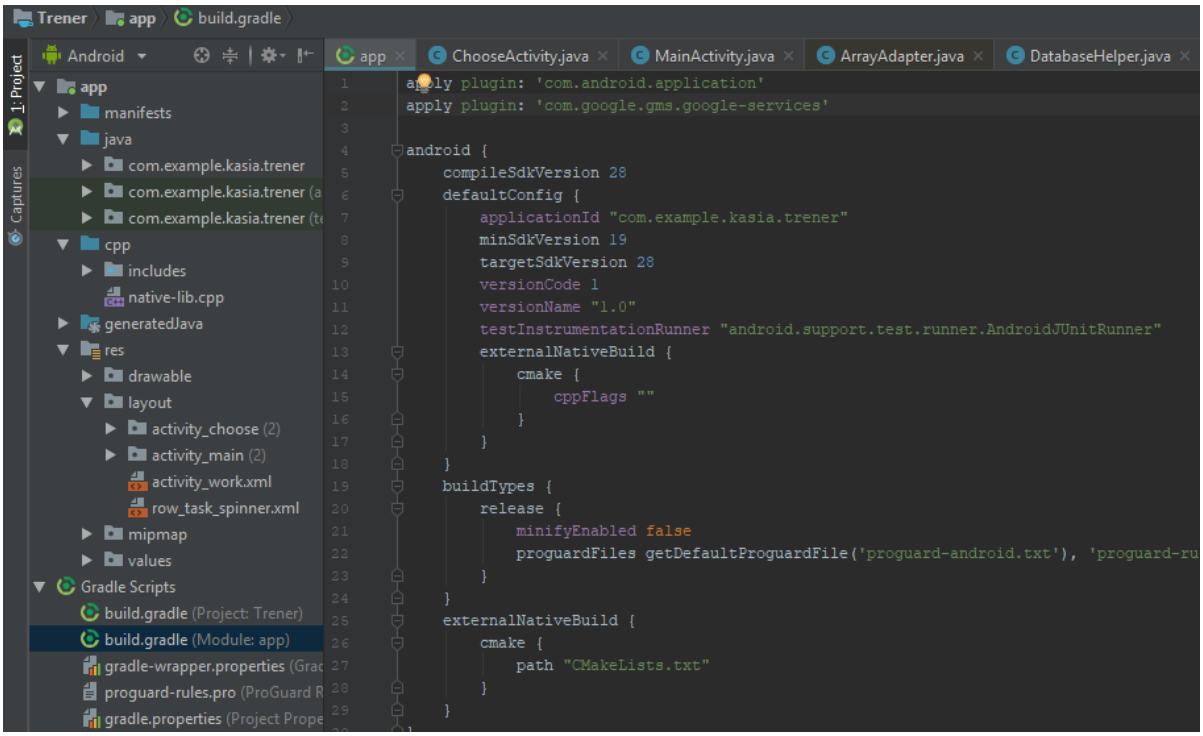
2. Należy podać nową nazwę dla projektu i zaakceptować warunki.

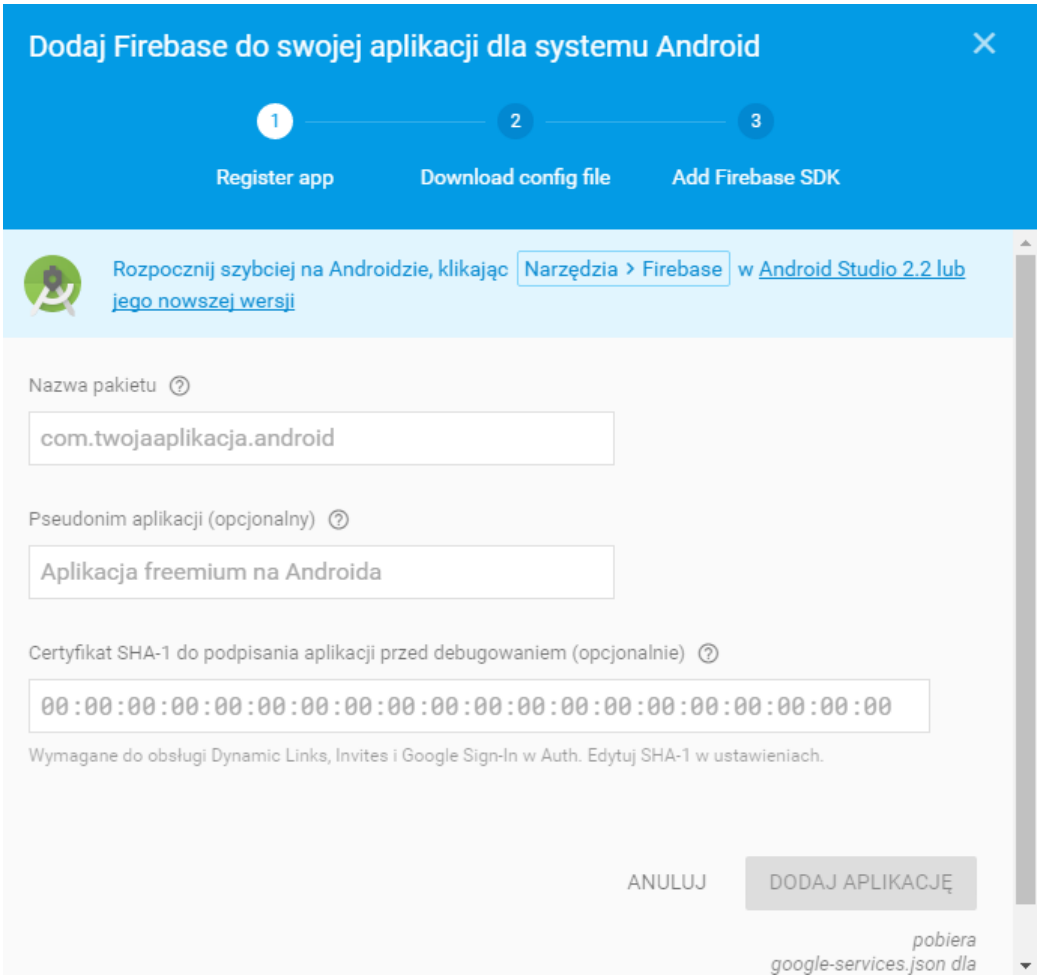


3. Ze strony Firebase niezbędny do konfiguracji jest plik google-services.json, dlatego trzeba go pobrać i umieścić w odpowiednim folderze aplikacji.



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.grandle jako applicationId.



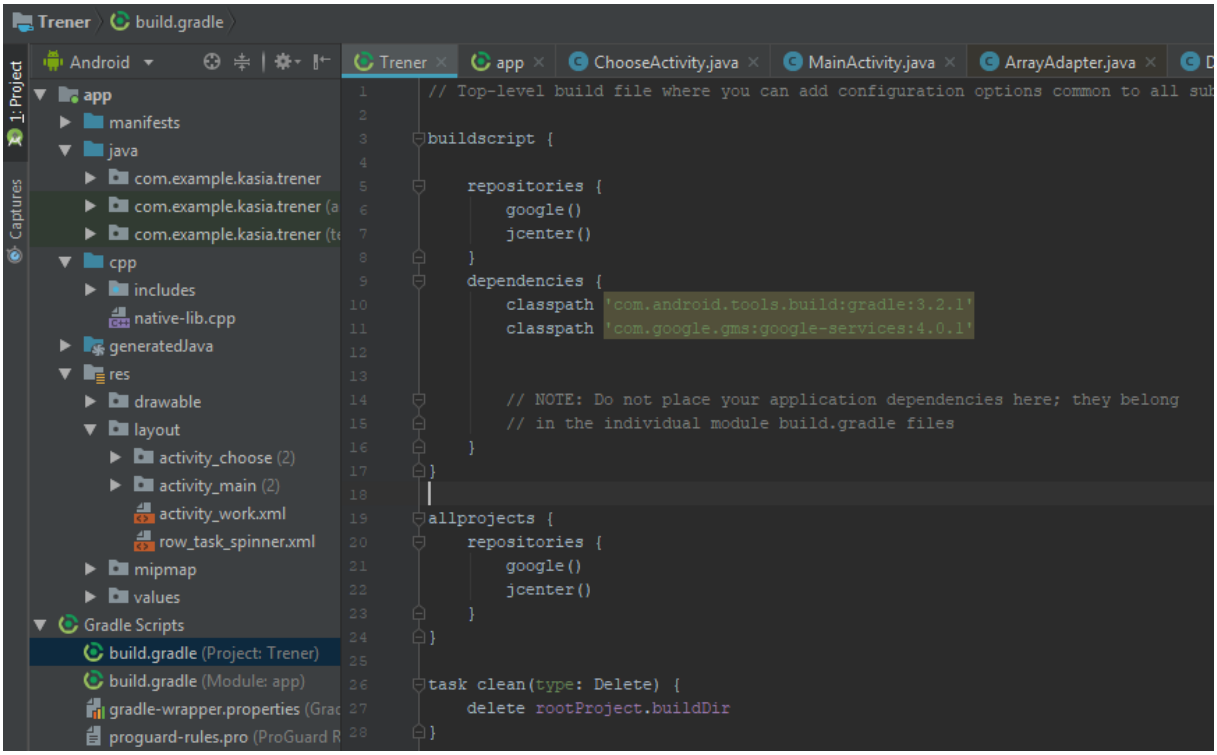


5. W tym samym pliku dopisać trzeba linijkę 2 oraz 40.

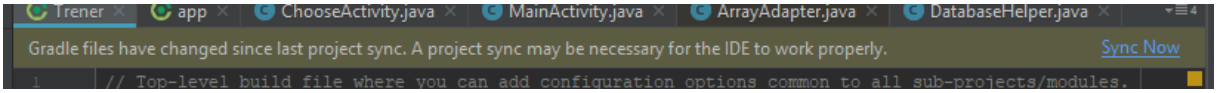
```
32  dependencies {
33      implementation fileTree(dir: 'libs', include: ['*.jar'])
34      implementation 'com.android.support:appcompat-v7:28.0.0'
35      implementation 'com.android.support.constraint:constraint-layout:1.1.3'
36      testImplementation 'junit:junit:4.12'
37      androidTestImplementation '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  }
```

```
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
```

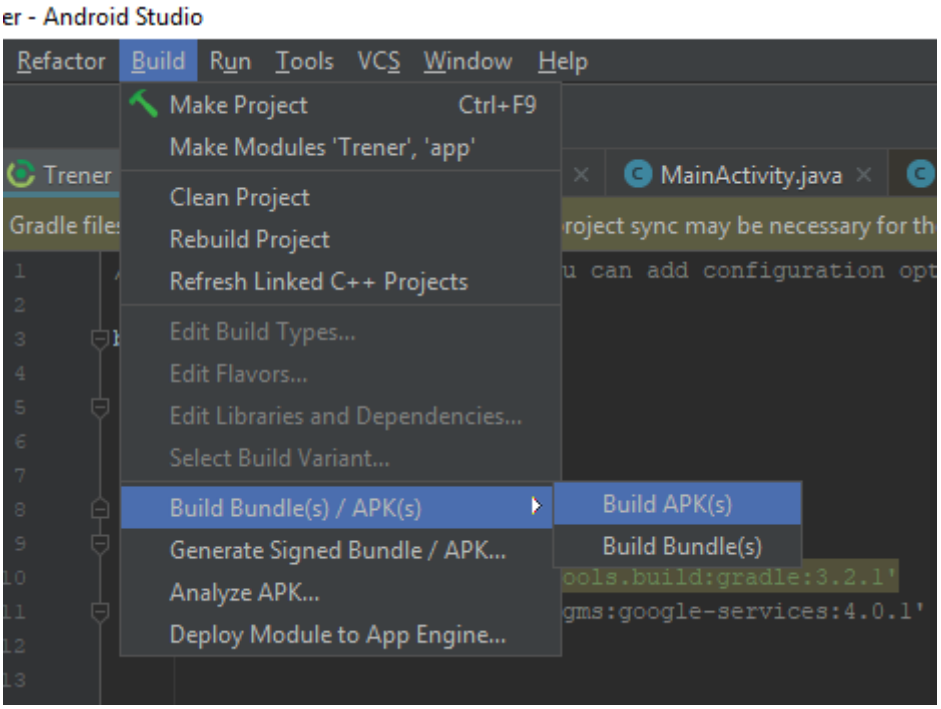
Do pliku `build.gradle` na poziomie aplikacji dodać należy także linijkę 11.



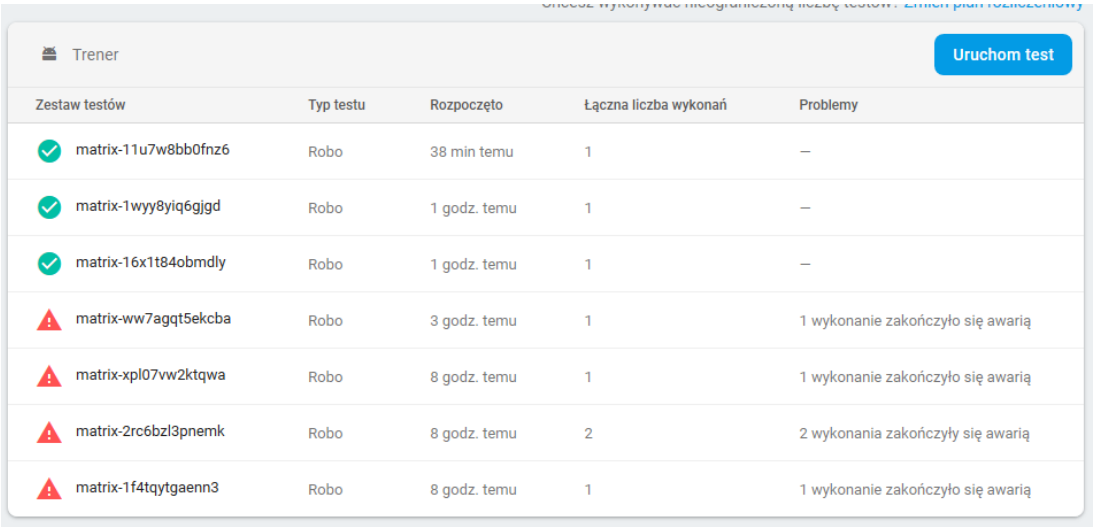
Gdy zostaną dodane powyższe linijki kodu oraz plik json zostanie przeniesiony do folderu aplikacji można przystąpić do synchronizacji projektu, co będzie łatwe dzięki pojawieniu się komunikatu zachęcającego do tego. Wystarczy nacisnąć „Sync Now”.



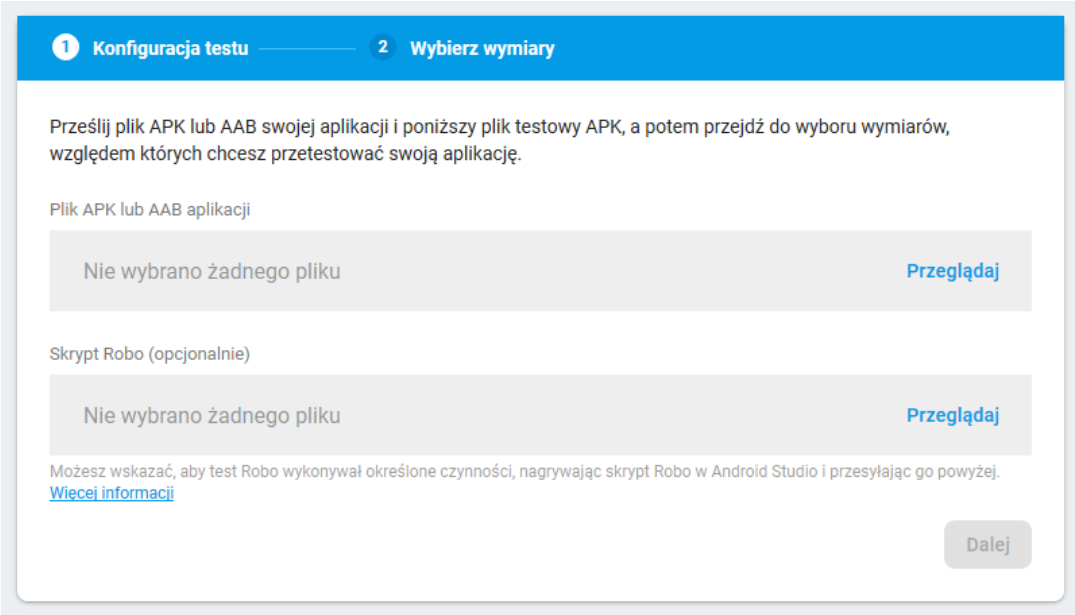
6. Z tak przygotowanej aplikacji konieczne jest teraz uzyskanie pliku apk, który następnie będzie wykorzystany przy testach. W Prawym dolnym rogu programu pojawi się komunikat o utworzeniu pliku i możliwości przejścia do jego lokalizacji.



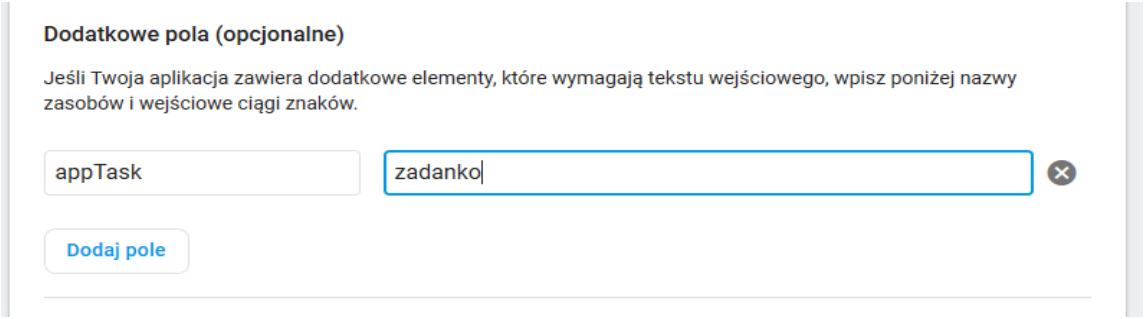
7. Najtrudniejsze zadanie zostało wykonane. Teraz wystarczy włączyć konsolę Firebase i wybrać „Uruchom test”.



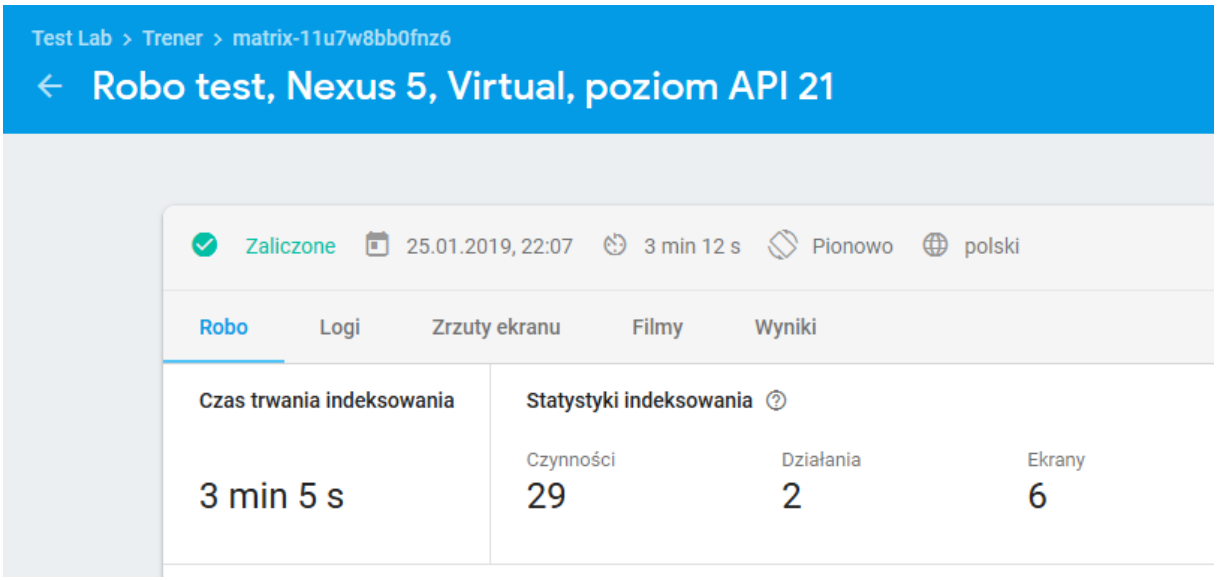
8. Plik apk, który wcześniej utworzono należy tutaj załadować i wybrać przycisk „Dalej”. Przeniesie on użytkownika do panelu wyboru urządzeń do testów, które zostały pokazane na samym początku.



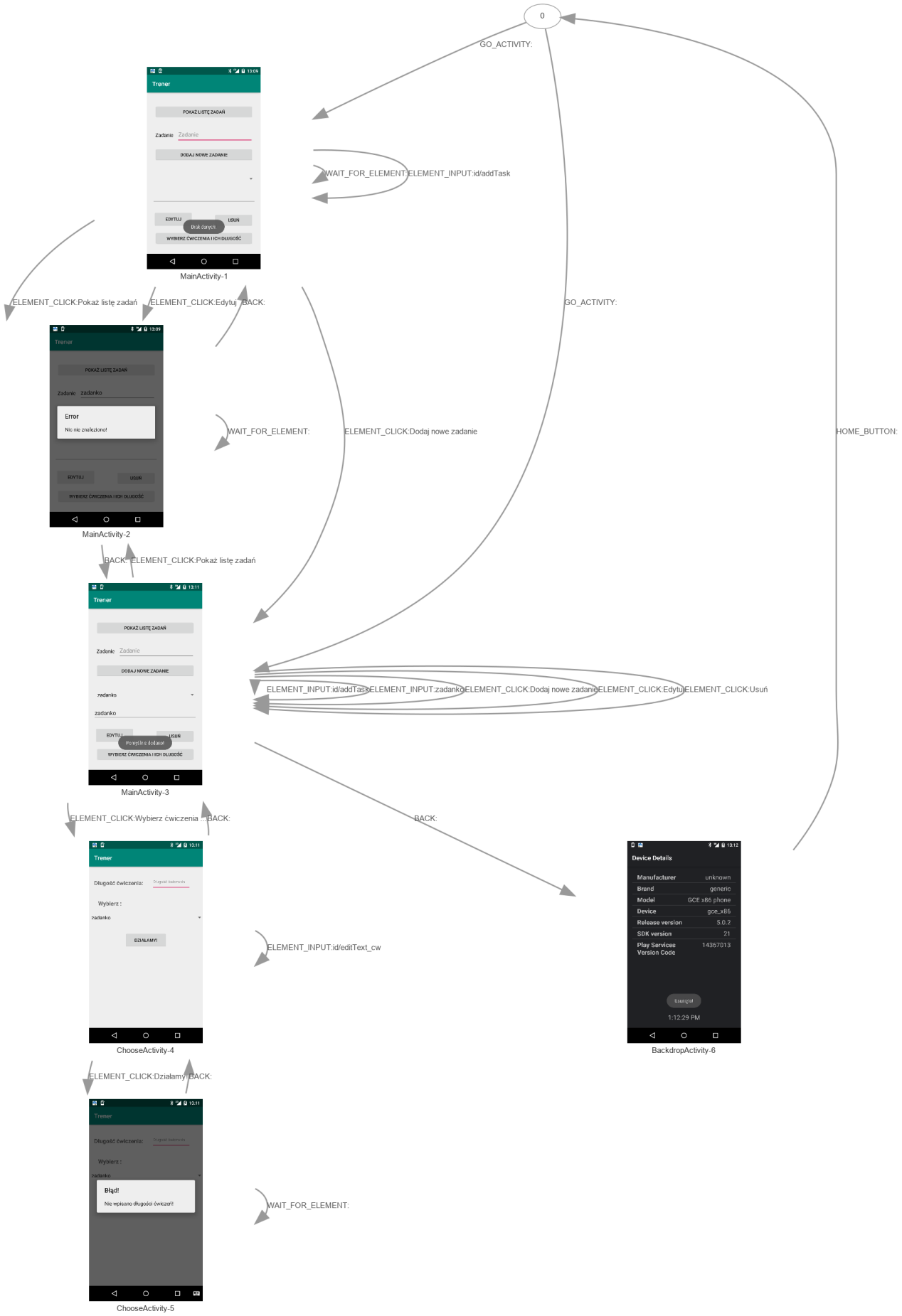
9. W ramach testu podano do Robota dane jednego z pól, które konkretnie ma wybrać do wpisania tekstu. Można tak zrobić z każdym polem, do którego jest dostęp po nazwie.



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



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.


0
GO_ACTIVITY:
MainActivity-1
WAIT_FOR_ELEMENT:ELEMENT_INPUT:id/addTask
ELEMENT_CLICK:Pokaż listę zadań
ELEMENT_CLICK:Edytuj
BACK:
GO_ACTIVITY:
MainActivity-2
WAIT_FOR_ELEMENT:
ELEMENT_CLICK:Dodaj nowe zadanie
HOME_BUTTON:
BACK:
ELEMENT_CLICK:Pokaż listę zadań
MainActivity-3
ELEMENT_INPUT:id/addTask
ELEMENT_INPUT:zadanko
ELEMENT_CLICK:Dodaj nowe zadanie
ELEMENT_CLICK:Edytuj
ELEMENT_CLICK:Usuń
ELEMENT_CLICK:Wybierz ćwiczenia
BACK:
BACK:
ChooseActivity-4
ELEMENT_INPUT:id/editText_cw
BackdropActivity-6
ELEMENT_CLICK:Działamy!
BACK:
ChooseActivity-5
WAIT_FOR_ELEMENT:

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.