Praca konkursowa

Odkryj Rudy

Autor: Bartłomiej Pacia, [barpac02@gmail.com](mailto:barpac02@gmail.com)

Opiekun: Donat Dubiel, knei.pracownia@gmail.com

Koło Naukowe Elektroniki i Informatyki w Knurowie

ul. Batorego 4, 44-194 Knurów

Table of Contents

[0. Wprowadzenie 1](#_Toc59150592)

[1. Przeznaczenie projektu 1](#_Toc59150593)

[2. Opis rozwiązania 1](#_Toc59150594)

[2.1. Zarys techniczny 1](#_Toc59150595)

[2.3 Funkcje serwerowe 2](#_Toc59150596)

[2.4 Turystyczna Baza Danych 3](#_Toc59150597)

[3. Wyniki 4](#_Toc59150598)

[3.1. Współpraca z Nadleśnictwem 5](#_Toc59150599)

[Współpraca z gminą Kuźnia Raciborska 5](#_Toc59150600)

[3.4 Odbiór 5](#_Toc59150601)

[4. Plany na przyszłość 5](#_Toc59150602)

[4.1 Powiększenie liczby regionów 5](#_Toc59150603)

[4.2 Program lojalnościowy 5](#_Toc59150604)

[4.3 Inne funkcje 5](#_Toc59150605)

[4.4 Udostępnienie bazy danych 6](#_Toc59150606)

[4.5 Nowa nazwa 6](#_Toc59150607)

### Wprowadzenie

Rudy to licząca ok. 3000 mieszkańców miejscowość położona w powiecie raciborskim, w gminie Kuźnia Raciborska. Ma bogatą historią, sięgającą 1258 roku, kiedy na tutejsze tereny przybyli Cystersi. Rudy są ważnym punktem na turystycznej mapie województwa śląskiego, znanym ze swoich walorów historycznych i przyrodniczych. Otoczone są Lasami Rudzkimi – jednym z największych kompleksem leśnym na Górnym Śląsku, bogatym w liczne gatunki drzew i zwierząt, pełnym uroczysk i innych ciekawych miejsc, do których dojechać można wytyczonymi ścieżkami rowerowymi.

### Shape, square Description automatically generatedPrzeznaczenie projektu

*Odkryj Rudy* to turystyczny system informacji o okolicy. W jego skład wchodzi aplikacja mobilna na iOS i Androida, serwer w chmurze, baza danych oraz zestaw narzędzi usprawniających różne etapy pracy z systemem. Pierwotnie przeznaczony był dla miejscowości Rudy i jej najbliższego otoczenia, w 2020 roku został rozszerzony na cały teren Gminy Kuźnia Raciborska.

Obecnie w aplikacji umieszczono ponad 70 miejsc ze zdjęciami, historią, ciekawostkami, współrzędnymi geograficznymi i innymi właściwościami. Są one podzielone według kategorii: *zabytki*, *ciekawe miejsca*, *atrakcje*, *kamienie pamiątkowe* i *świetlice wiejskie*. Użytkownicy mogą dodawać do miejsc polubienia (popularne „lajki”) oraz komentarze ze zdjęciami, a jeśli są w pobliżu miejsca (bliżej niż 100 m), mają możliwość zaznaczenia miejsca jako *odkryte*. Dodatkowo administrator systemu może wysyłać na urządzenia z zainstalowaną aplikacją powiadomienia, aby np. poinformować o wydarzeniu kulturalnym.

Rys. 1 - obecna ikona aplikacji

strona internetowa: <https://odkryjrudy.pl>

### Opis rozwiązania

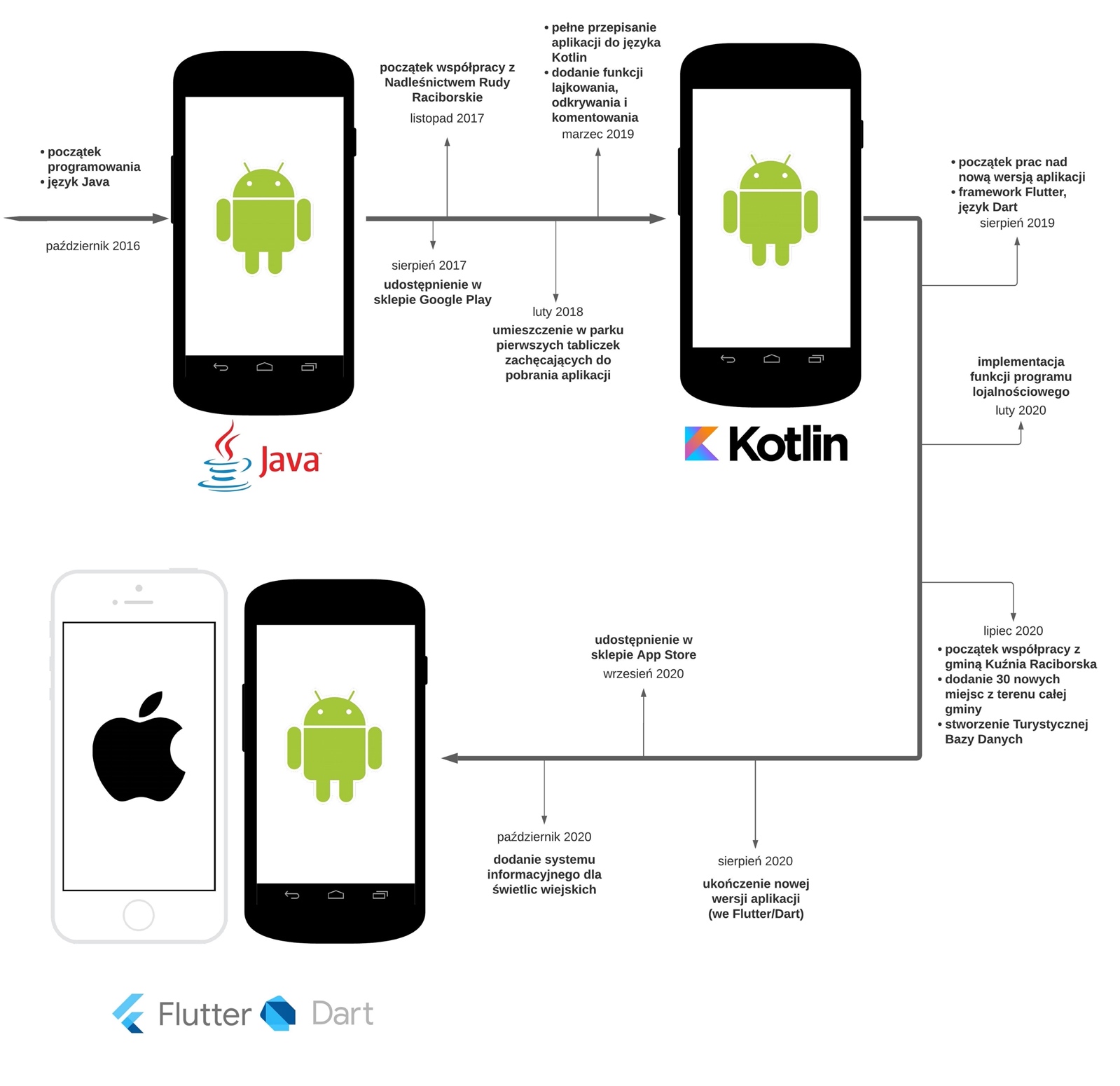
### Zarys techniczny

*Odkryj Rudy* składa się z 3 głównych komponentów:

* aplikacji mobilnej
* funkcji serwerowych
* Turystycznej Bazy Danych

Projekt był i jest rozwijany na zasadzie jakiej????

**2.2 Aplikacja mobilna**

Aplikacja mobilna była pierwszym (i przez długi czas jedynym) komponentem systemu, a jednocześnie tym, który przeszedł najwięcej zmian (zob. rys. 2). Obecnie napisana jest w całości w języku programowania Dart we frameworku Flutter. Największą zaletą takiego rozwiązania jest możliwość stworzenia wydajnego, natywnego programu na iOS i Androida z jednego kodu źródłowego.

Rys. 2 – wybrane wydarzenia z procesu rozwijania aplikacji

### 2.3 Funkcje serwerowe

*Odkryj Rudy* nie posiada własnego serwera w rozumieniu tradycyjnym – zamiast niego, do uruchamiania kodu serwerowego korzysta z usługi Google Cloud Functions. Pozwala ona na uruchamianie kodu serwerowego w chmurze, tj. na serwerach automatycznie zarządzanych przez Google. Pozwala to zminimalizować koszta i czas wymagane na zarządzanie własnym serwerem. Upraszcza też wiele kwestii związanych z bezpieczeństwem i czynnościami utrzymaniowymi (aktualizowanie oprogramowania, optymalizowanie wydajności).

Funkcje były początkowe napisane w języku JavaScript, ale z powodu jego wad (brak statycznego typowania) potęgowanych przez szybko rosnącą ilość kodu zostały przepisane do TypeScript.

Funkcje serwerowe w *Odkryj Rudy* można podzielić na 2 rodzaje:

* *triggery* – ich zadaniem jest uruchamianie kodu w odpowiedzi na jakąś akcję podjętą przez użytkownika (np. dodanie komentarza ze zdjęciem – funkcje sprawdzają, czy komentarz nie jest wulgarny i kompresują zdjęcie)

**LUCIDCHART DODAWANIE ZDJĘCIA**

* *funkcje zaplanowane* – ich zadaniem jest przeprowadzanie czynności utrzymaniowych – np. operacji, które muszą być uruchamiane w konkretnych interwałach (np. odświeżanie jakości powietrza co 15 minut, pozbywanie się zdjęć będących częścią usuniętych komentarzy)

Użytkownicy *Odkryj Rudy* mogą tworzyć kilka rodzajów obiektów – polubienia miejsc, wizyty oraz komentarze (wraz z załączonymi do nich zdjęciami). Wszystkie te dane przechowywane są w Cloud Firestore – nierelacyjnej bazie danych (noSQL) zapewnianej przez platformę Google Firebase. Głównymi zaletami *Cloud Firestore* jest możliwość nasłuchiwania na zmiany i odpytywania bazy w czasie rzeczywistym. Dane umieszcza się naprzemiennie w kolekcjach i dokumentach, co swoją strukturą przypomina drzewo systemu plików.

Wadą takiego rozwiązania jest konieczność zastosowania redundancji danych. Jednak korzyści (elastyczność, możliwość odpytywania i nasłuchiwania bazy danych w czasie rzeczywistym), jakie daje użycie bazy noSQL jak Cloud Firestore przewyższają wady:

* zwiększone zapotrzebowanie na miejsce (które jest tanie)
* problemy związane z duplikacją danych/redundancją danych (które zostały rozwiązane dzięki użyciu funkcji zaplanowanych).

### 2.4 Turystyczna Baza Danych

Jest to najmłodsza część systemu, stworzona w lecie 2020 roku. Jej zadaniem jest ustandaryzowanie formatu przechowywania danych oraz ogólne usprawnienia procesu pracy z nimi. Kluczowym pojęciem w Bazie jest *paczka*. *Paczka* to po prostu folder o określonej strukturze zawierający zdjęcia, teksty i inne dane (np. pliki JSON) opisujące dany rejon (np. gminę).

Paczka to pojęcie trochę abstrakcyjne, ale dobrze oddające jej istotę. Tak naprawdę istnieją 2 rodzaje paczek:

1. *Paczka źródłowa* to ściśle określona struktura folderów i plików (drzewo), zawierająca wszystkie zebrane dane o danym rejonie (zdjęcia w wysokiej jakości, teksty w kilku językach)
2. *Paczka danych* to ogólne określenie na folder lub plik .zip zawierający tekstowe i graficzne dane, gotowe do użycia przez aplikację. Paczka danych może być

Dlaczego to rozróżnienie pomiędzy *paczkami źródłowymi* a *paczkami* *danych?*

Paczka źródłowa może zawierać np. 10 zdjęć jakiegoś miejsca w wysokiej jakości. Jednak bezpośrednie użycie tych 10 zdjęć w aplikacji mobilnej oznaczałoby marnotrawienie zasobów użytkowników (transfer mobilny, pamięć), którzy byliby zmuszeni do ich pobierania na swoje urządzenia. Dlatego tworzy się *paczkę danych*. Paczki danych to mniejsze, skompresowane wersje *paczek źródłowych*. która zawierają np. tylko 4 ze wspomnianych 10 zdjęć, i to mocno skompresowane. Ich jakość będzie zupełnie wystarczająca do wyświetlenia na smartfonie, a jednocześnie będą zajmować od kilku do kilkunastu razy mniej pamięci.

Z *paczki źródłowej* można stworzyć *paczkę danych* za pomocą programów zawartych w projekcie *database-tools*.

Zawiera on kilka prostych programów upraszczających cały proces pracy z bazą danych (zob. rys. 3)

Diagram

Description automatically generated

* program *generate* generuje paczkę danych. Przyjmuje kilka argumentów, z których najważniejsze to ID paczki oraz język, w którym będą treści tekstowe paczki danych (oczywiście o ile takowe przetłumaczone teksty są dostępne w paczce źródłowej).Po uruchomieniu program *chodzi* po drzewie folderów i plików paczki źródłowej (jego struktura znajduje się w pliku drzewo\_paczki\_1.txt), wybiera odpowiednie zdjęcia i pliki tekstowe i tworzy z nich folder paczki danych.
* program *compress* kompresuje paczkę danych z postaci folderu wyjściowego do pliku .zip
* program *upload* przesyła paczkę danych w formacie .zip do Cloud Storage oraz odpowiednie metadane do Cloud Firestore. Natychmiast po przesłaniu stają się one widoczne dla użytkowników aplikacji.

W grudniu 2020 roku *database-tools* zostały przepisane z Pythona do języka Go, dzięki czemu wzrosła niezawodność i szybkość działania programu. Dodatkowo struktura kodu została gruntownie zrefaktoryzowana, dzięki czemu dodawanie nowych funkcji do programów w przyszłości będzie prostsze.

### Wyniki

Można powiedzieć, że projekt odniósł sukces. Do końca grudnia 2020 roku aplikację pobrało ponad 2000 użytkowników[[1]](#footnote-1).

### A picture containing text, grass, outdoor, wooden Description automatically generatedWspółpraca z Nadleśnictwem

W 2017 roku nawiązana została współpraca z Nadleśnictwem Rudy Raciborskie. Nadleśnictwo zamieściło na swojej stronie informację o aplikacji oraz zakupiło i przekazało 50 aluminiowych tabliczek z nadrukowanym kodem QR, zachęcających do pobrania aplikacji. Obecnie większość z nich jest zamieszczona przy ciekawych miejscach w przyklasztornym parku oraz na całym terenie Lasów Rudzkich (zob. rys. 4).

### Współpraca z gminą Kuźnia Raciborska

Rys. 3 – tablica informacyjna z przykręconym kodem QR i zachętą do pobrania aplikacji.

W 2020 roku nawiązana została współpraca z gminą Kuźnia Raciborska (zob. załącznik gmina.jpg). Obszar zawarty w aplikacji rozszerzony został na teren całej gminy. Aplikacja otrzymała również wsparcie finansowe, dzięki czemu możliwy był zakup licencji w Apple Developer Program ($99 rocznie) i opublikowanie aplikacji w sklepie App Store. Do systemu dodany został również prosty system informacyjny, który ułatwia komunikację pomiędzy instruktorami świetlic wiejskich (czyli osobami, które je prowadzą) i osobami które korzystają z oferty świetlic wiejskich. Nie był on jednak jeszcze używany w praktyce, gdyż został ukończony akurat, gdy zaczęła się druga fala pandemii i wszystkie wiejskie świetlice zostały zamknięte.

### 3.4 Odbiór

Aplikacja jest pozytywnie odbierana przez użytkowników, o czym świadczy średnia ocen 4,9/5 oraz pozytywne recenzje w sklepie Google Play. W załącznikach (pliki kom\_\*.jpg) znajduje się kilka wybranych recenzji, a plik *mail.png* zawiera ciekawy e-mail, jaki otrzymałem od najaktywniejszego użytkownika aplikacji po poproszeniu go o podzielenie się opinią i wrażeniami.

### 4. Plany na przyszłość

*Odkryj Rudy* jest dojrzałym projektem, zbudowanym na stabilnym stosie technologicznym, a architektura całego systemu dobrze sprawdza się w praktyce. Stwarza to szerokie pole do rozwoju aplikacji.

### Powiększenie liczby regionów

Oczywistym polem do rozwoju dla aplikacji typu *Odkryj Rudy* jest poszerzenie obszaru, który w sobie zawiera. Struktura *paczki danych* jest prosta, a zarazem efektywna – wystarczy zabrać dane (zdjęcia ciekawych miejsc i zabytków danego regionu, ich lokalizacje i historię), a następnie umieścić je w odpowiedniej strukturze folderów i plików – dzięki czemu program *generate* z *database-tools* będzie w stanie stworzyć *paczkę danych* gotową do użycia przez aplikację. Cechą wyróżniającą Odkryj Rudy jest otwarty format *paczki*. Dzięki temu będzie mieć szansę na stanie się tanim i dobrym wyborem dla gmin/rejonów, które podobnie jak Rudy 4 lata temu, nie mają zdigitalizowanej bazy danych swojej oferty turystycznej.

### Program lojalnościowy

Ciekawym pomysłem jest wdrożenie programu lojalnościowego. W ramach tego programu użytkownicy mogliby odbierać określone nagrody za odkryte miejsca. Na przykład, jeśli użytkownik odkrył 10 miejsc, to w Punkcie Informacji Turystycznej w Rudach może za darmo odebrać album ze zdjęciami, a jeśli jest zapalonym turystą i odkrył np. 50 miejsc,

Prace nad programem lojalnościowym od strony technicznej zostały zakończone na początku 2020 roku. W toku były rozmowy z Nadleśnictwem Rudy Raciborskie dot. szczegółów nagród i kwestii prawnych. Niestety, z powodu pandemii i innych wydarzeń realizacja programu lojalnościowego w ramach Nadleśnictwa stała się niemożliwa. Dlatego też zwróciłem się z tą kwestią do gminy Kuźnia Raciborska i otrzymałem pozytywną odpowiedź. Prace trwają.

### Inne funkcje

* **automatyczne odkrywanie miejsc**

Po włączeniu odpowiedniej opcji, aplikacja będzie automatycznie miejsce jako odkryte, gdy użytkownik jest w pobliżu.

* **stworzenie wersji internetowej**

Użytkownicy mieliby dostęp do większości funkcji aplikacji mobilnej z poziomu przeglądarki internetowej. Stanie się to gdyż framework Flutter w przyszłości pozwoli na budowanie do wersji webowej.

* **dodanie testów jednostkowych i integracyjnych**

Obecnie to ja jestem jedną osobą pracującą nad systemem i znam go „na wylot”, więc brak testów nie jest dużym problemem. Jednak w przypadku, gdy w przyszłości nad *Odkryj Rudy* pracowałoby parę osób, testy znacząco uproszczą proces rozwijania aplikacji i dadzą programistom pewność, że kod, który piszą, działa.

* **kompletna przebudowa części serwerowej**

Zrezygnowanie z funkcji serwerowych na rzecz „normalnego” REST API uruchamianego architektury serwerowej w kierunku wykorzystania konteneryzacji (Docker, Kubernetes), aby w (odległej i niepewnej) przyszłości móc efektywnie skalować liczbę instancji poszczególnych komponentów w zależności od obciążenia.

### Udostępnienie bazy danych

Jako wielki entuzjasta oprogramowania open-source uważam, że każdy mieszkaniec powinien mieć wolny i darmowy dostęp do wszystkich danych o swojej okolicy, w możliwie najprostszy sposób. Dlatego ważną dla mnie kwestią jest publiczne udostępnienie Turystycznej Bazy Danych (projekty *database* i *database-tools*). Na razie nie jest jeszcze to możliwe z powodu szczątkowej dokumentacji oraz kwestii prawnych, które wymagają przeanalizowania.

### Nowa nazwa

Pomniejszą, ale ważną z marketingowego punktu widzenia jest nazwa aplikacji i całego systemu. Obecnie pełna, oficjalna nazwa systemu to *Odkryj Rudy i okolice*, ale na potrzeby tego tekstu pozwoliłem sobie skracać ją do *Odkryj Rudy.*

Gdy proponowane wyżej ulepszenia wejdą w życie (w szczególności, gdy w aplikacji dostępna będzie większa liczba regionów), nazwa *Odkryj Rudy* przestanie być odpowiednia. Aby wybrać nową nazwę, potrzeba dobrze znać zakres projektu, by nie być zmuszonym do kolejnego rebrandingu w krótkim czasie. Na razie jednak *Odkryj Rudy i okolice* spełnia swoje zadanie.

**5. Źródła**

Jako że projekt jest nieustannie rozwijany od 4 lat, niemożliwym jest wymienienie wszystkich wykorzystanych źródeł, materiałów i bibliotek. Jednak nieocenioną pomocą były oficjalne dokumentacje języków programowania i narzędzi programistycznych, z których korzystałem.

strona internetowa i dokumentacja systemu Android <https://developer.android.com>

strona internetowa i dokumentacja platform firmy Apple <https://developer.apple.com>

strona internetowa i dokumentacja frameworka Flutter <https://flutter.dev>

strona internetowa i dokumentacja języka programowania Dart <https://dart.dev>

strona internetowa i dokumentacja języka programowania Kotlin <https://kotlinlang.org>

strona internetowa i dokumentacja języka programowania TypeScript <https://www.typescriptlang.org>

strona internetowa i dokumentacja języka programowania Go <https://golang.org>

strona internetowa i dokumentacja platformy Google Firebase <https://firebase.google.com>

strona internetowa i dokumentacja Google Cloud Platform <https://cloud.google.com>

forum dla programistów StackOverflow <https://stackoverflow.com>

Ze względu na dużą ilość informacji zgromadzoną w Turystycznej Bazie Danych, nie wszystkie zgromadzone tam informacje mają źródła (źródła są wymienione ogólnie, ale nie konkretnie).

1. Źródło: usługa Firebase Analytics i sklep Google Play. [↑](#footnote-ref-1)