

**PROJEKT INŻYNIERSKI**

**„Temat pracy”**

**Karol Kadłubowski**

**Nr albumu 290366**

**Kierunek: Informatyka**

**Specjalność: Bazy Danych i Inżynieria Systemów**

**PROWADZĄCY PRACĘ**

**Dr inż. Michał Sawicki**

**Katedra Systemów Rozproszonych i Urządzeń Informatyki**

**Wydział Automatyki, Elektroniki i Informatyki**

**OPIEKUN, PROMOTOR POMOCNICZY (jeśli został powołany)**

**<stopień naukowy oraz imię i nazwisko>**

**GLIWICE Rok 2022**

Spis treści

[Rozdział 1 Wstęp 1](#_Toc98759118)

[Rozdział 2 [Analiza tematu] 3](#_Toc98759119)

[Rozdział 3 [Wymagania i narzędzia] 5](#_Toc98759120)

[Rozdział 4 [Właściwy dla kierunku – np. Specyfikacja zewnętrzna] 7](#_Toc98759121)

[Rozdział 5 [Właściwy dla kierunku – np. Specyfikacja wewnętrzna] 9](#_Toc98759122)

[Rozdział 6 Weryfikacja i walidacja 11](#_Toc98759123)

[Rozdział 7 Podsumowanie i wnioski 13](#_Toc98759124)

[Bibliografia 15](#_Toc98759125)

[Spis skrótów i symboli 19](#_Toc98759126)

[Źródła 20](#_Toc98759127)

[Lista dodatkowych plików, uzupełniających tekst pracy 21](#_Toc98759128)

[Spis rysunków 22](#_Toc98759129)

[Spis tablic 23](#_Toc98759130)

# Rozdział 1 Wstęp

* wprowadzenie w problem/zagadnienie
* Osadzenie problemu w dziedzinie
* Cel pracy
* Zakres pracy
* Zwięzła charakterystyka rozdziałów
* Jednoznaczne określenie wkładu autora, w przypadku prac wieloosobowych – tabela z autorstwem poszczególnych elementów pracy

Pierwsza linia akapitu z wcięciem. Czcionka Times New Roman lub Callibri 12pt. Obustronne wyrównanie. Interlinia 1.3

# Rozdział 2 – Analiza Tematu

# Temat projektu

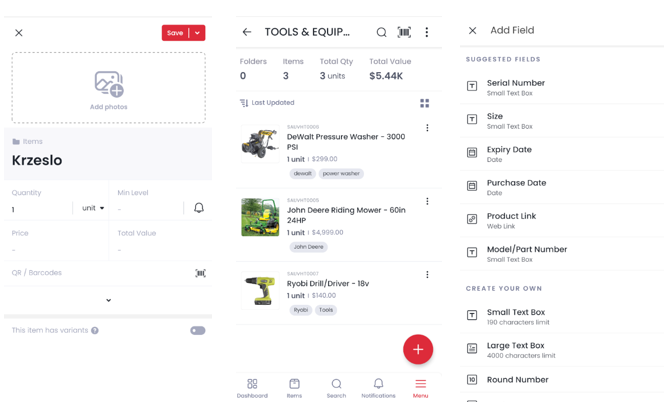
W ramach projektu autora ma zostać zaprojektowana aplikacja do przeprowadzania inwentaryzacji z wykorzystaniem zewnętrznej drukarki kodów kreskowych 1D oraz 2D. Projekt ten obejmuje połączenie dwóch niezwiązanych pozornie rodzin aplikacji: rodziny aplikacji inwentaryzacyjnych oraz rodziny aplikacji służących do obsługi urządzeń zewnętrznych. Na podstawie obecnie już istniejących rozwiązań rynkowych, praca ta udowadnia innowacyjność podejścia autora do opisywanego problemu.

# Aplikacje inwentaryzacyjne

Wśród aplikacji ze Sklepu Play, dominują dwa rodzaje tego typu aplikacji. Pierwszy z nich odnosi się do aplikacji prowadzących spis środków trwałych. Księgowane przedmioty charakteryzują się długim czasem użytkowania. Drugim typem aplikacji inwentaryzacyjnych są aplikacje prowadzące spis produktów sklepowych. Ten rodzaj aplikacji skupia się bardziej na biznesowych aspektach księgowania towaru. Poniżej autor opisuje sztandarowe aplikacje reprezentujące te dwie grupy.

## Sortly – inwentaryzacja środków trwałych

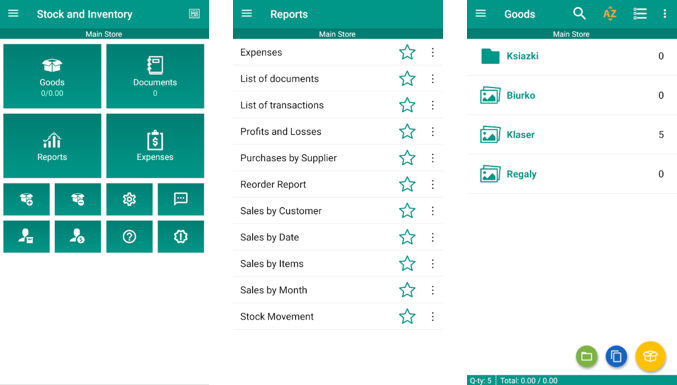
Aplikacja ta podręcznikowo pokazuje funkcjonalności dotyczące omawianej grupy, jednak z racji jej wszechstronności, można za jej pomocą prowadzić również inwentaryzacje sklepowe. Trzeba zwrócić uwagę na jej estetyczny wygląd, który czerpie z wyglądu wielkich aplikacji jak Facebook, Instagram, itp.. Aplikacja wymaga logowania na własne konto – może być ono sparowane z Facebookiem czy Google. Wszystkie dane wprowadzane na dane konto będą za jego pośrednictwem przechowywane w chmurze. Aplikacja została zrealizowana w modelu ograniczonej bezpłatnej wersji, możliwej do płatnego rozszerzenia do wersji premium. Początkowa konfiguracja zakłada podanie przez użytkownika docelowego typu prowadzonej inwentaryzacji (np. domowa, magazynowa, sklepowa). Tak jak każda aplikacja inwentaryzacyjna, oferuje ona możliwość zapisywania wyposażenia oraz jego szczegółowych cech. Użytkownik może tworzyć swoje własne cechy. Poszczególne przedmioty mogą zostać zgrupowane w foldery co ułatwia ich segregacje. Ze szczególnych typów informacji, autor wyszczególnia możliwość ustawienia ilości przedmiotów, dodawania zdjęć oraz przypisania kodu kreskowego 1D lub 2D. Każdy przedmiot posiada również zapis przebiegu swojej historii. Aplikacja umożliwia filtrowanie oraz sortowanie przedmiotów według zróżnicowanych kryteriów. Krytyczną funkcjonalnością ułatwiającą wyszukiwanie, jest wykorzystywanie aparatu w celu skanowania kodów kreskowych. Zeskanowanie kodu może również służyć działaniom jak zmiana ilości przedmiotów lub nadanie znacznika. Aplikacja pozwala również na import z plików tekstowych oraz na eksport do plików tekstowych i PDF. Jest dostępna również opcja generowania raportów (np. raporty związane z brakującymi towarami lub przeprowadzonymi transakcjami). Najważniejsze cechy wersji premium odnoszą się do możliwości dodawania nieskończonej ilości przedmiotów, przypisywaniu kodów kreskowych oraz na ustawianiu spersonalizowanych powiadomień.



Rys. 1. Zrzuty ekranu z aplikacji Sortly

## Stock and Inventory Simple – inwentaryzacja towaru w sklepie

Aplikacja nie wymaga zakładania własnego konta – wszystkie dane zapisywane są bezpośrednio na urządzeniu. Istnieje jednak możliwość składowania danych na dysku Google. Większość funkcji jest bezpłatnych, jednak księgowanie cen zostaje uruchomione dopiero po wykupieniu wersji premium. Po pierwszym uruchomieniu ukazuje się ekran samouczka połączony z możliwościami konfiguracji. Aplikacja oczywiście pozwala na zapis przedmiotów oraz ich cech, jednak ilość cech jest bardzo ograniczona. Jak w poprzedniej aplikacji, istnieje możliwość ustawienia ilości przedmiotów, dodawania zdjęć oraz przypisania kodu kreskowego 1D lub 2D. Prowadzony jest również zapis historii przedmiotu. Możliwe jest tworzenie folderów, co w połączeniu z systemem zarządzania, przypominającym ten w menadżerze plików urządzenia, pozwala na bardzo łatwe przerzucanie i kopiowanie list przedmiotów. Aplikacja umożliwia również filtrowanie oraz sortowanie przedmiotów według zróżnicowanych kryteriów. Możliwa jest również identyfikacja przedmiotu poprzez zeskanowanie kodu kreskowego. Aplikacja pozwala na import oraz eksport do plików tekstowych. Wszystkie powyższe wymienione cechy w większości odpowiadają tym, opisywanym w poprzedniej aplikacji. Jednak rzeczą, która wyróżnia tą aplikację, są możliwości wizualizacji przepływów towaru. Najważniejszą zaletą jest generowanie bardzo szczegółowych raportów, odpowiadających kryteriom biznesowym (np. raport wydatków, sprzedaży miesięcznej, zysków i strat). Raporty te zawierają czytelne dane w postaci tabeli oraz posiadają możliwość narysowania wykresu. Wygenerowany raport można otworzyć jako plik PDF lub plik excelowy – z możliwością zapisu. Inną funkcją jest dodawanie dostawców, klientów oraz sklepów, którzy mają związek z ruchem towarów podlegających inwentaryzacji.



Rys. 2. Zrzuty ekranu z aplikacji Stock and Inventory Simple

## Porównanie aplikacji

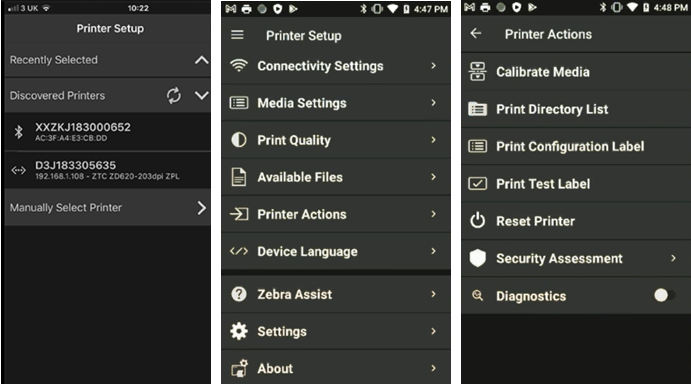
Obie aplikacje posiadają podobne funkcjonalności, jednak pierwsza skupia się dużo bardziej na wszechstronności, oferując szeroką gamę opcji dostosowania zapisywanych przedmiotów. Natomiast druga aplikacja zakraja te możliwości, dając jednak możliwość interpretacji wyników raportów generowanych na podstawie wprowadzanych danych.

# Aplikacje do obsługi urządzeń zewnętrznych

Kolejnym bardzo ważnym aspektem jest dziedzina aplikacji służących do obsługi drukarki kodów kreskowych oraz tagów RFID. Aplikacje służą te zarówno do konfiguracji początkowej, jak i utrzymania kontaktu z urządzeniem.

## Zebra Printer Setup Utility

Aplikacja ta wykorzystuje bluetooth żeby wyszukiwać urządzenia w pobliżu. Następnie za jej pomocą można sparować drukarkę z telefonem – można też to zrobić za pomocą interfejsu systemowego. Ustawienia łączności umożliwiają konfiguracje kanałów łączności takich jak bluetooth, wifi czy nfc. Ustawienia mediów odpowiadają za opcje związane z drukowaniem etykiet. W tej zakładce można zmieniać rozmiar etykiety oraz wybrać czy druk ma być realizowany w formacie etykiet lub paragonów. W przypadku druku etykiet, można ustawić opcje rozpoznawania, w którym miejscu rozpoczyna się kolejna etykieta. Można też dostosować dodatkowe przesunięcie, ułatwiające łatwiejsze odrywanie etykiety. Istnieje także opcja zmiany typu wydruku (np. termotransferowy lub termalny). W zakładce dotyczącej jakości druku, ustawiana jest wyrazistość wydruku. Tutaj także można wykonać wydruki testowe. W zakładce „dostępne pliki” możemy wybrać z urządzenia plik do wydrukowania, który zostanie przesłany do drukarki. Zakładka „akcje drukarki” umożliwia kalibracje drukarki (według zapisanych ustawień mediów), oraz wydrukowanie: etykiety z zapisem folderów i plików z pamięci wewnętrznej, etykiety z konfiguracją urządzenia czy etykiety testowej. Możliwy jest także zdalny reset drukarki. Ostatnią opcją konfiguracyjną jest zmiana języka odbioru drukarki – za pomocą tego języka będą interpretowane pliki wysyłane do drukarki (np. zwykły tekst, ZPL, CPCL).



Rys. 3. Zrzuty ekranu z aplikacji Zebra Printer Setup Utility

## Ocena aplikacji

Aplikacja posiada bardzo wiele opcji konfiguracji drukarki, jednak nie wszystkie oferowane funkcje działają na posiadanej przez autora drukarce. Możliwe że wynika to z drobnych różnic między drukarkami, które wszystkie obsługiwane są za pomocą jednej zbiorczej aplikacji. Kolejną wadą jest brak możliwości wgrywania własnych czcionek do drukarki. Jest to o tyle ważne, że bazowe czcionki nie posiadają obsługi np. polskich znaków. Z racji tych braków autor nie mógł przeprowadzić całkowicie zdalnej konfiguracji urządzenia. Z tego powodu została ona wykonana poprzez połączenie z komputerem.

# Rozdział 3 – Wymagania i narzędzia

# Wymagania funkcjonalne aplikacji

Na podstawie opisu rozwiązań dostępnych na rynku, autor nakreśla ogólny zamysł funkcji zawartych w jego aplikacji. Z racji tego, że głównym elementem pracy inżynierskiej autora jest biblioteka do obsługi urządzeń zewnętrznych, autor skupi się na rozszerzeniu funkcjonalności z tym związanych oraz na zapewnieniu podstawowych funkcjonalności dotyczących przeprowadzania inwentaryzacji.

## Zapisywanie danych o przedmiotach podlegających inwentaryzacji

Głównym przeznaczeniem aplikacji będzie gromadzenie danych o realnie istniejących przedmiotach. Będzie się na to również składało na zapisywanie jak największej ilości cech, by jak najlepiej oddać naturę przedmiotu.

## Obsługa czynności związanych z księgowaniem tych przedmiotów

W związku z celem przeprowadzania inwentaryzacji aplikacja będzie posiadać ułatwienia dotyczące zarządzaniem przedmiotów oraz ich dodawaniem. Nieodzowną tutaj częścią będzie wykorzystanie drukarki jak i skanera kodów.

## Obsługa połączenia bluetooth dla drukarki ??

Do zadań aplikacji będzie należało zarządzaniem stanu połączenia bluetooth telefonu. Aplikacja będzie wyszukiwać pobliską drukarkę oraz wykonywać opcje parowania. Wszystkie te kroki mają na celu zapewnienie ciągłej komunikacji telefonu z urządzeniami zewnętrznymi.

## Konfigurowanie ustawień drukarki ??

Aplikacja powinna być w stanie zainstalować wszelkie niezbędne ustawienia na drukarce, niezbędne do prawidłowego drukowania kodów. Będzie się na to składać: ustawienie języka drukowania, kalibracja drukarki, ustawienie przerwy ułatwiającej odrywanie etykiet, wybranie sposobu detekcji nowej etykiety podczas przewijania rolki, wgrywanie odpowiedniej czcionki będącej w stanie obsługiwać natywne znaki.

## Drukowanie kodów kreskowych 1D, 2D

W celu przeprowadzania efektywnej inwentaryzacji, często na przedmiotach jej podlegających umieszcza się różnego rodzaju znaczniki. Podstawowym powodem takiego działania jest to, że podczas kolejnych inwentaryzacji, czas potrzebny na zaksięgowanie danego przedmiotu zostaje drastycznie skrócony. Dzieje się tak przez to, że dużo łatwiej zidentyfikować tak opisany przedmiot – poprzez wyszukanie danych z etykiety czy chociażby poprzez zeskanowanie widniejącego na niej kodu. Przewidywany okres użytkowania etykiet kodów kreskowych jest estymowany na okres jednego roku. Jest to spowodowane tym, że obecna oferta rynkowa drukarek mobilnych obejmuje tylko i wyłącznie drukowanie z wykorzystaniem technologii druku termicznego. Ten rodzaj druku nie wymaga tuszu, a co za tym skutkuje, drukarki te posiadają mniejsze rozmiary. Papier etykietowy jest pokryty cienką warstwą substancji chemicznej, która przy kontakcie z głowicą drukarki wywołuje reakcję, skutkującą zabarwieniem papieru. Ten typ druku nie jest jednak przeznaczony do wieloletniego użytkowania – etykiety te pod wpływem czynników zewnętrznych takich jak temperatura, wilgoć, naświetlenie czy uszkodzenia mechaniczne ulegają zatraceniu swojej czytelności. Alternatywą byłoby zastosowanie urządzenia obsługującego druk termotransferowy. Takie etykiety mają zdecydowanie wyższą odporność na czynniki zewnętrzne oraz dłuższy czas użytkowania. Niestety urządzenia tego typu nie spełniają mobilnych kryteriów aplikacji, ponieważ z racji wykorzystania taśmy termotransferowej, ich gabaryty są często 2-3 razy większe niż w przypadku drukarek termicznych. Autor rozważał rozwiązanie tego problemu poprzez zaprojektowanie specjalnego rodzaju nosidełek pozwalających na komfortowe przenoszenie większych drukarek.

## Skanowanie tekstu ze zdjęcia

Aplikacja posiada funkcje rozpoznawania tekstu na zdjęciach. Jest to umotywowane tym, że przedmioty podlegające inwentaryzacji (przed wprowadzaniem wprowadzonego przez autora systemu) często są oznakowane swoim kodem w postaci napisu na obudowie. Aplikacja ta oszczędza czasu użytkownikowi wpisywania każdego z kodów. Będzie to mogło zostać wykonane poprzez zrobienie zdjęcia, a następnie zaznaczenie obszaru rozpoznawania tekstu.

# Technologie

Technologie użyte w projekcie nie ograniczają się jedynie do narzędzi programistycznych, ale także do ogólnego zarządzania projektem. Autor stara się implementować metodyki stosowane w przedsiębiorstwach, jako że są one szeroko przyjęte jako standard i gwarantują optymalną ścieżkę rozwoju aplikacji. Innym aspektem jest też wybór drukarki. W tej kwestii autor ma do dyspozycji drukarkę produkcji firmy Zebra, z uwagi na swoją służbową współpracę w ramach projektu.

## System operacyjny

Zgodnie z założeniami projektowymi, autor zdecydował się na zaprojektowanie aplikacji pod natywny system Androida. Decyzja ta została podyktowana dostępnością bibliotek obsługujących połączenie z drukarką. Dodatkowo autor korzysta z wbudowanych urządzeń telefonu jak np. Bluetooth. Obsługa tego typu urządzeń może różnić się w zależności od wersji systemu urządzenia, a takie niuanse są najdokładniej kontrolowane w technologii natywnej dla danego typu urządzenia.

## Język programowania

Do wyboru w dziedzinie Androida są dwa języki programowania: Java lub Kotlin. Autor zdecydował się na wybranie języka Kotlin z następujących powodów:

* Kotlin w 2017 roku stał się oficjalnym językiem Androida, co oznacza ciągłe wsparcie oraz rozwój technologii,
* Kotlin obsługuje zestaw narzędzi Jetpack Compose. Jest to nowoczesny standard wykorzystujący deklaratywny interfejs użytkownika oraz techniki programowania reaktywnego.

W przypadku wykorzystania Javy autor musiałby konstruować widoki za pomocą języka znaczników XML. Nie możliwym byłoby też wykorzystanie najnowocześniejszych rozwiązań implementowanych przez język Kotlin.

## Wykorzystane biblioteki kluczowych funkcjonalności

W celu komunikacji z drukarką, w projekcie wykorzystywany jest pakiet bibliotek firmy Zebra o nazwie ZSDK\_ANDROID\_API. Jest to jedyne dostępne rozwiązanie na rynku, które zapewnia kompleksową obsługę funkcji dedykowanego urządzenia. Skanowanie obrazu przeprowadzane jest za pomocą biblioteki Google ML Kit. Alternatywnym rozwiązaniem byłoby użycie biblioteki Firebase ML Kit.

## Środowisko programistyczne

W związku z wykorzystaniem Jetpack Compose, dedykowanym środowiskiem do konstruowania tego rodzaju aplikacji jest Android Studio firmy JetBrains. Środowisko to zapewnia wszystkie niezbędne funkcje związane z tworzeniem aplikacji mobilnych. Jego najważniejszymi funkcjami są:

* zarządzanie estetyką kodu,
* debugowanie kodu,
* wbudowany emulator,
* możliwość podglądu pamięci telefonu,
* możliwość śledzenia wydajności aplikacji,
* współpraca z systemem kontroli wersji,
* bezpośrednie wgrywanie lub generowanie instalatorów, mające na celu instalacje aplikacji na prawdziwym urządzeniu.

## System kontroli wersji

W ramach systemu kontroli wersji zdecydowałem się na używanie Githuba. Oprócz zapisywania historii zmian projektu pełni on dodatkowe funkcje, które porządkują systematyczną pracę. Alternatywą tutaj byłby np. Bitbucket w połączeniu z Jirą, jednakże Bitbucket posiada on ograniczenia dotyczące ilości czasu obliczeniowego.

### Zapis historii zmian plików

Dzięki połączeniu z systemem kontroli wersji każda zmiana kodu jest zapisywana wraz z nagłówkiem oraz znacznikiem czasowym. Dzięki temu można dokładnie prześledzić kolejność wprowadzanych zmian oraz ewentualnie przez kogo zostały one wprowadzone. Autor jako dobrą praktykę przyjął wrzucania małych partii kodu, dotyczących danego zagadnienia. Takie podejście ułatwia później postępowanie, w przypadku gdy jedna z funkcjonalności powoduje błąd. Wtedy bardzo prosto można cofać kolejne zmiany, szukając w którym momencie pojawił się dany błąd. Dzięki temu poprawa będzie odbywać się na minimalnym fragmencie kodu, jednocześnie nie ingerując w kod dotyczący innych zagadnień.

### Tablica zadań

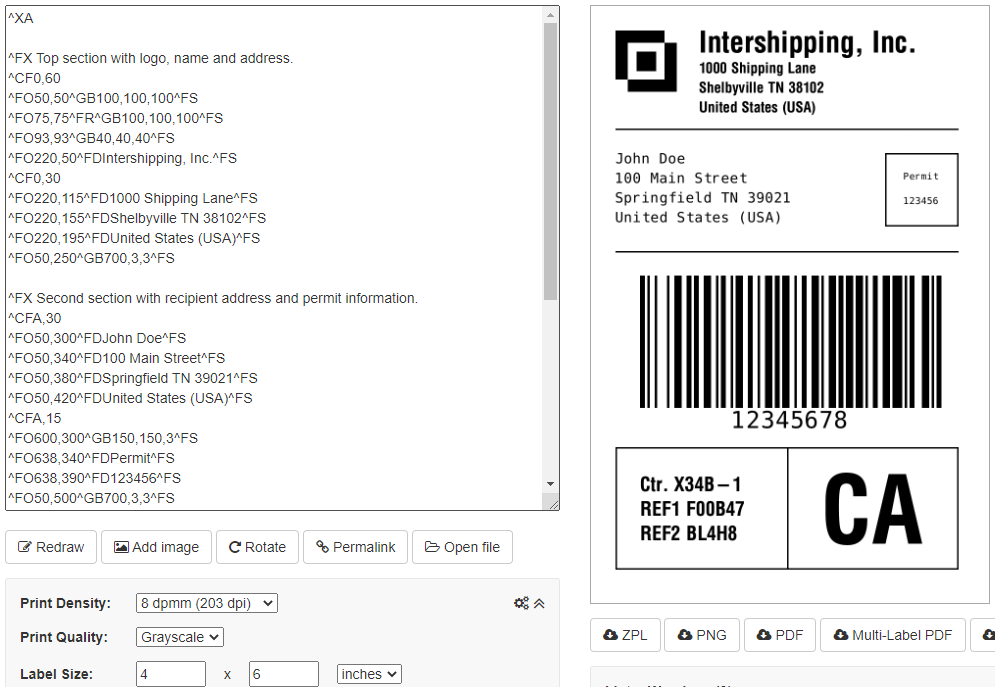
Github oferuje możliwość zapisywania konkretnych zadań, które odnoszą się do kolejnych funkcjonalności aplikacji lub korekty tych już zaimplementowanych. W ramach takiej tablicy, można na początku utworzyć szkielet, według którego aplikacja będzie otrzymywała nowe funkcjonalności. Poszczególne zadania posiadają swój priorytet oraz mniej lub bardziej szczegółowy opis, by programista jak najlepiej mógł oddać ideę powierzonego zadania. W wygodny sposób można też utworzyć nową gałąź dotyczącą danego zadania. Dzięki temu różne osoby mogą pracować nad zadaniami w innych częściach projektu, jednocześnie nie wpływając na główny rdzeń kodu. System kontroli wersji zapobiega też niezgodnościom przy łączeniu gałęzi. Przy łączeniu, programista zostanie poproszony o rozwiązanie konfliktów, jeśli okaże się, że dany fragment kodu został zmieniony na dwóch niezależnych gałęziach.

### Automatyzacja testów

Github oferuje również możliwość wykorzystania mocy obliczeniowej serwerowych maszyn obliczeniowych, w celu automatycznego przeprowadzania testów kodu. Zgodnie z dobrymi praktykami programistycznymi, około 80% całego kodu, a w szczególności część logiczna powinna zostać pokryta testami jednostkowymi. Większe moduły mogą być testowane za pomocą testów integracyjnych oraz akceptacyjnych. Możliwym jest, by testy te były automatycznie przeprowadzane za pomocą systemu kontroli wersji. Następowałoby to w momencie łączenia dwóch gałęzi. Warunkiem niezbędnym do takiego złączenia, byłoby pozytywne przejście wcześniej zaimplementowanych testów. W przypadku niepowodzenia, programista zostanie powiadomiony, że dana funkcjonalność jest niesprawna, a to uniemożliwi dołączenie niesprawnego kodu do głównej gałęzi projektu.

## Technologia drukowania

W celu komunikacji z drukarką, autor wybrał język ZPL. Za jego pomocą można wykonywać polecenia konfiguracyjne drukarki lub programować wygląd etykiet. Dodatkowo jest dostępny internetowy kreator etykiet Laberary, dzięki któremu istnieje możliwość podglądu widoku etykiety, przy jej programowaniu.



Rys. 4. Internetowy kreator etykiet Labelary - [Labelary Online ZPL Viewer](http://labelary.com/viewer.html)

# Rozdział 4 [Właściwy dla kierunku – np. Specyfikacja zewnętrzna]

Jeśli to specyfikacja zewnętrzna:

* wymagania sprzętowe i programowe
* sposób instalacji
* sposób aktywacji
* kategorie użytkowników
* sposób obsługi
* administracja systemu
* kwestie bezpieczeństwa
* przykład działania
* scenariusze z systemu (ilustrowane zrzutami z ekranu lub generowanymi dokumentami)

|  |
| --- |
|  |
| Rys.4.1. Podpis rysunku jest pod rysunkiem |

# Rozdział 5 – Specyfikacja wewnętrzna

# Architektura aplikacji

W aplikacji została zastosowana architektura MVVM (Model-View-ViewModel). Opiera się na zastosowaniu trzech logicznych warstw. Ma to na celu rozdzielenie odpowiedzialności oraz zmniejszenie zależności pomiędzy odpowiednimi warstwami.



Rys. 5. Diagram architektury MVVM

## Model

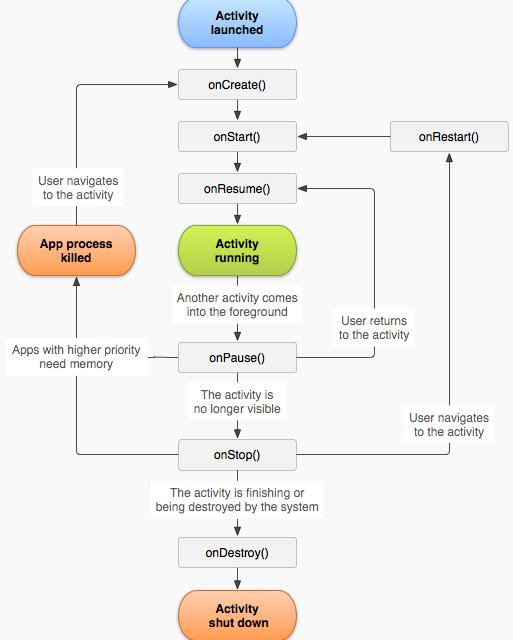
Warstwa modelu jest najniższą warstwą odpowiedzialną za dostęp do danych, w której znajdują się klasy odzwierciedlające domenę, czyli obszar tematyczny, do którego przynależy aplikacja. W warstwie modelu znajdują się m. in repozytoria, które odpowiadają za zwracanie danych z jednego lub kilku źródeł. Funkcjonują również serwisy, które operują na zadanych danych. Znajdują się też tutaj poszczególne przypadki użycia (ang. use-case). W bardzo czytelny sposób wyznaczają one funkcjonalności aplikacji.

## Widok

Warstwa ta odpowiada za graficzną reprezentację oraz interakcję z użytkownikiem. Należy zaznaczyć, że to widok ma referencję do modelu, a nie odwrotnie. To znaczy, że wszystkie dane z modelu są przekazywane do widoku.

## Model widoku

Zadaniem tej warstwy jest przesłanie danych do widoku oraz wymiana informacji z modelem. Innymi słowy, oznacza to pobieranie danych, które to skutkuje aktualizacją widoku. W Androidzie dodatkowo warstwa ta nabiera zupełnie innej wagi. W związku z tym, że wszystkie ekrany aplikacji Androida budowane są w aktywnościach, podlegają one własnemu cyklowi życia. Z tej racji dane zapisywane bezpośrednio w warstwie widoku, podlegają usunięciu przy zakończeniu cyklu życia aktywności. Dlatego też wszystkie dane przechowywane są w viewmodelu, ponieważ jest on odporny na zmiany cyklu życia aktywności. W ten sposób aplikacja jest chroniona przed utratą danych z powodu np. zablokowania telefonu czy zminimalizowania aplikacji.



Rys. 6. Diagram cyklu życia aktywności.

# Opis struktur danych

W aplikacji została zastosowana biblioteka Room, która pełni rolę abstrakcyjnej warstwy nad bazą danych SQLite. SQLite implementuje silnik SQL, dając możliwość używania bazy danych bez konieczności uruchamiania osobnego procesu RDBMS. Zaletą tej bazy jest wysoka wydajność przy obsłudze jednego użytkownika (którym jest nasza aplikacja), w porównaniu z na przykład MySQL czy PostgreSQL.

# Przegląd ważniejszych klas

W aplikacji główne funkcjonalności implementowane są za pomocą serwisów oraz przypadków użycia. W tym punkcie autor chciałby opisać najważniejsze z tych klas.

## Serwis bluetooth

Zadaniem serwisu jest wymiana informacji z drukarką. W celu nawiązania komunikacji, niezbędne jest obsłużenie poniższych funkcjonalności:

* obsłużenie permisji związanych z wykorzystaniem bluetootha,
* uruchomienie bluetooth na telefonie,
* wyszukanie pobliskich urządzeń,
* pobranie listy sparowanych urządzeń,
* przesłanie pliku etykiety pod zadany adres.

## Serwis etykiety

Etykiety projektowane są w języku Zebra Programming Language (ZPL). W tym serwisie konstruowany jest wygląd etykiety. Z racji przyjęcia przez autora, że aplikacja będzie wykorzystywana w Polsce, ustawione zostaje wsparcie polskich znaków specjalnych. Serwis też odpowiada za rozdzielenie poszczególnych elementów etykiety, w taki sposób, by żaden z nich nie był nieczytelny lub nie został wydrukowany poza fizyczną etykietą. Serwis ustala też jaki typ kodu ma zostać wydrukowany: kod typu 128 lub kod QR. W przypadku kiedy numer inwentarzowy okazałby się bardzo długi, serwis wymusi wydrukowanie kodu QR. Jest to spowodowane tym, że kody kreskowe wydłużają się proporcjonalnie do długości zakodowanej informacji i odpowiednio długi kod mógłby nie zmieścić się na etykiecie.

## Serwis podziału linii

Forma graficzna etykiety projektowana jest w języku ZPL. Język ten nie posiada jednak funkcji automatycznego rozdzielania wyrazów pomiędzy liniami. Z racji tego, żeby w czytelny sposób przekazać własności danego obiektu, wymagane jest ręczne rozdzielenie wyrazów pomiędzy kolejnymi liniami na etykiecie. Zadaniem tego serwisu jest rozdzielenie ciągu znaków pomiędzy zadaną ilość linii, ale w taki sposób, by w kolejnych liniach znajdowały się pełne wyrazy.

## Serwis odczytywania regexa

W ustawieniach aplikacji jest możliwe podanie przykładowego numeru inwentarzowego. Serwis ten odpowiada za translacje tego kodu na wyrażenie typu regex. Regex ten jest wykorzystany przy skanowaniu numerów inwentarzowych za pomocą kamery.

Jeśli to specyfikacja wewnętrzna:

* przedstawienie idei
* architektura systemu
* opis struktur danych (i organizacja baz danych)
* komponenty, moduły, biblioteki, przegląd ważniejszych klas (jeśli występują)
* przegląd ważniejszych algorytmów (jeśli występują)
* szczegóły implementacji wybranych fragmentów, zastosowane wzorce projektowe
* diagramy UML

krótka wstawka kodu w linii tekstu jest możliwa, np. **descriptor**, a nawet **descriptor\_gaussian**. Dłuższe fragmenty lepiej jest umieszczać jako rysunek, np. kod na rysunku 5.1, a naprawdę długie fragmenty – w załączniku.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | **package** polsl.iinf.lab;  **import** java.util.Random;  **public** **class** Main {  **public** **static** **void** main(String[] args) {  Random r = **new** Random();  // drawing a number from the range 1..10  **int** a = r.nextInt(10 + 1);  System.*out*.println(a);  // drawing a number from the range -5..15  System.*out*.println(r.nextInt(21) - 5);  }  } |

Rysunek 5.1: Pseudokod

# Rozdział 6 Weryfikacja i walidacja

* sposób testowania w ramach pracy (np. odniesienie do modelu V)
* organizacja eksperymentów
* przypadki testowe, zakres testowania (pełny/niepełny)
* wykryte i usunięte błędy
* opcjonalnie wyniki badań eksperymentalnych

|  |  |
| --- | --- |
| Tabela 4.2. Nagłówek tabeli jest nad tabelą. | |
| Poziom 1 | 24 pt |
| Poziom 2 | 20 pt |
| Poziom 3 | 16 pt |
|  | |

# Rozdział 7 Podsumowanie i wnioski

* uzyskane wyniki w świecie postawionych celów i zdefiniowanych wyżej wymagań
* kierunki ewentualnych danych prac (rozbudowa funkcjonalna …)
* problemy napotkane w trakcie pracy

# Bibliografia

1. Imię Nazwisko, Imię Nazwisko. *Tytuł książki*. Wydawnictwo, Warszawa, 2017.
2. Imię Nazwisko, Imię Nazwisko. Tytuł artykułu w czasopiśmie. *Tytuł czasopisma*, 157(8):1092–1113, 2016.
3. Imię Nazwisko, Imię Nazwisko, Imię Nazwisko. Tytuł artykułu konferencyjnego. *Nazwa konferencji*, str. 5346–5349, 2006.
4. Autor, jeśli znany. https: [www.adres.strony](http://www.adres.strony) (dostęp:dzień.miesiąc.rok)
5. <https://programistabyc.pl/mvvm-1-koncepcja/> (13.12.2022)
6. Alexandru Dumbravan, Clean Android Architecture, Pack Publishing, 2022.
7. <https://www.geeksforgeeks.org/android-build-a-movie-app-using-retrofit-and-mvvm-architecture-with-kotlin/> (13.12.2022)

Dodatki

# Spis skrótów i symboli

*DNA* kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)

*MVC* model – widok – kontroler (ang. *model–view–controller*)

*N* liczebność zbioru danych

µ stopień przynależności do zbioru

# Źródła

Jeżeli w pracy konieczne jest umieszczenie długich fragmentów kodu źródłowego,

należy je przenieść do tego miejsca.

# Lista dodatkowych plików, uzupełniających tekst pracy

W systemie, do pracy dołączono dodatkowe pliki zawierające:

* źródła programu,
* dane testowe
* film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
* itp.

# Spis rysunków

4.1 Podpis rysunku jest pod rysunkiem 12

5.1 Pseudokod w listings 14

5.2 Pseudokod w minted 14

# Spis tablic

6.1 Opis tabeli nad nią 16