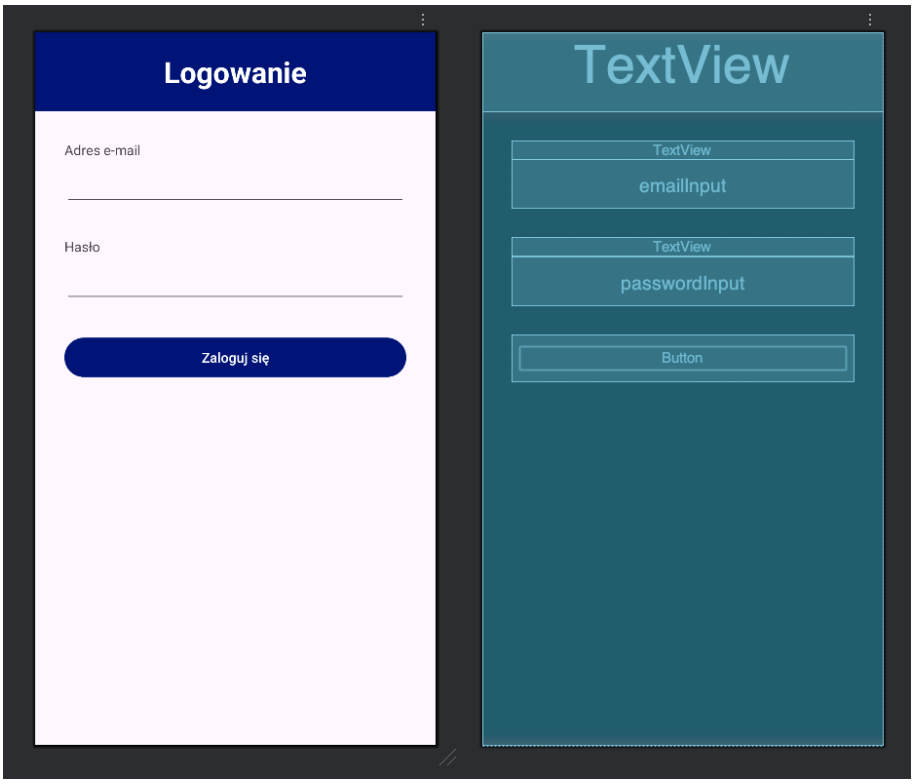
Zadanie podsumowujące – Android

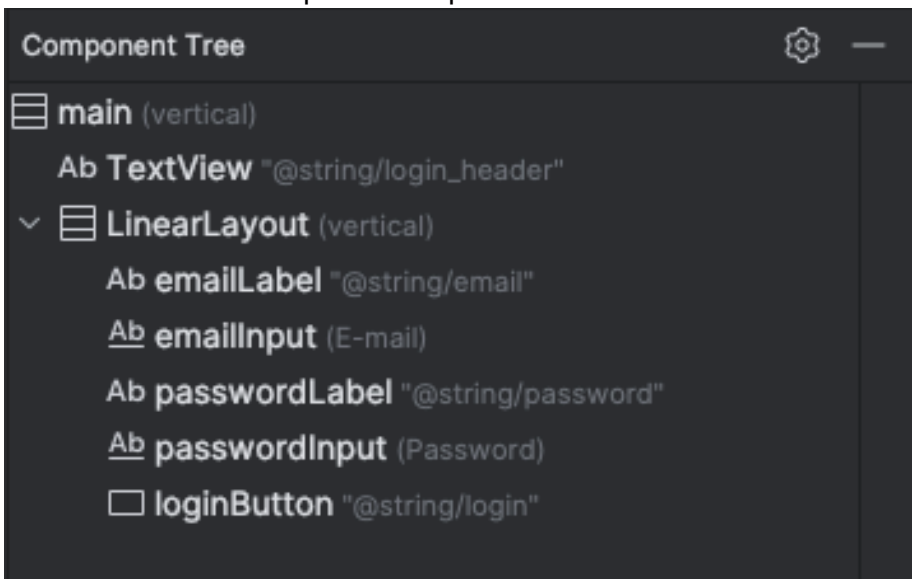
Wykonaj aplikację dla systemu Android zgodnie z poniższymi wymaganiami.
Rozwiązanie zadania umieść w repozytorium na Github.
Działanie przykładowej aplikacji pokazano na nagraniu załączonym w plikach zadania.
Wymagane grafiki zostały dołączone w plikach zadania.

1. Utwórz nowy projekt w Android Studio. Projekt bazuje na ***Empty Views Activity***.
    a. Nazwa projektu: **Zadanie Podsumowujące <Imię> <Nazwisko>**
    b. Nazwa pakietu: ***edu.zsk.<nazwisko>***
2. Aplikacja składa się z następujących plików:
    a. Pliki Java:
        i. Aktywności:
            1. ***LoggedInActivity.java***
            2. ***MainActivity.java***
            3. ***NotificationActivity.java***
        ii. Fragmenty:
            1. ***AppDialogFragment.java***
            2. ***FirstFragment.java***
            3. ***SecondFragment.java***
    b. Pliki layoutu:
        i. ***activity_logged_in.xml***
        ii. ***activity_main.xml***
        iii. ***activity_notification.xml***
        iv. ***fragment_first.xml***
        v. ***fragment_second.xml***
3. Wymagania dotyczące layoutu ***activity_main.xml***.
   Widok przedstawia zrzut ekranu:



Wymagania dla widoku:

a. Struktura drzewa komponentów przedstawiona na zrzucie ekranu:



b. Wymagania dotyczące stylowania:
   i. Nagłówek z tekstem „Logowanie”:
      1. Wysokość: ***dopasowanie do zawartości***
      2. Szerokość: ***dopasowanie do rodzica***
      3. Margines wewnętrzny: ***20dp***
      4. Ułożenie tekstu: ***do środka***
      5. Rozmiar tekstu: ***30sp***
      6. Kolor tła: ***#FF001577***
      7. Kolor tekstu: ***#FFFFFFFF***
      8. Styl testu: ***pogrubiony***
   ii. Zagnieżdżony element LinearLayout:

1. Wysokość: ***dopasowanie do rodzica***
2. Szerokość: ***dopasowanie do rodzica***
3. Margines wewnętrzny: ***30dp***

iii. Element emailLabel:
1. Label dla elementu emailInput
2. Szerokość: ***dopasowanie do rodzica***
3. Wysokość: ***dopasowanie do zawartości***

iv. Element emailInput:
1. Szerokość: ***dopasowanie do rodzica***
2. Wysokość: ***dopasowanie do zawartości***
3. Podpowiedzi do automatycznego podpowiadania: ***emailAddress***
4. EMS: ***10***
5. Minimalna wysokość: ***50sp***
6. Margines dolny: ***30dp***
7. Typ pola: ***adres e-mail***

v. Element passwordLabel:
1. Label dla elementu passwordInput
2. Szerokość: ***dopasowanie do rodzica***
3. Wysokość: ***dopasowanie do zawartości***

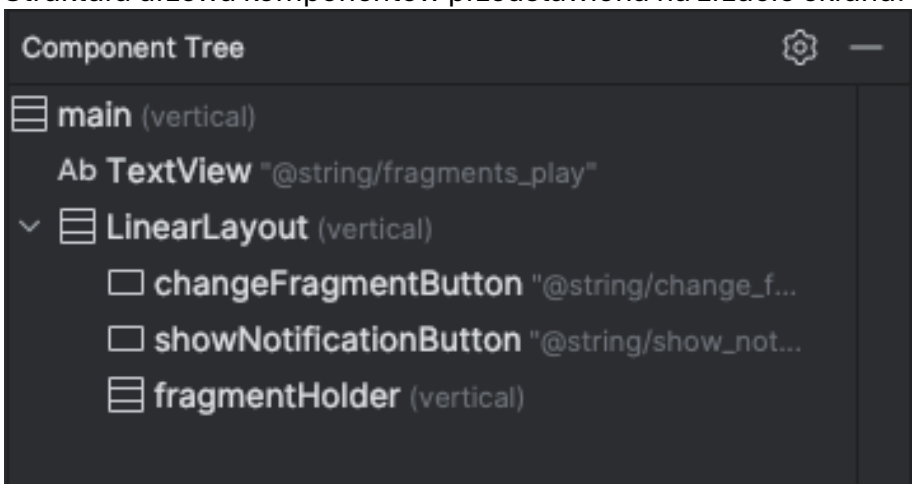
vi. Element passwordInput:
1. Szerokość: ***dopasowanie do rodzica***
2. Wysokość: ***dopasowanie do zawartości***
3. Podpowiedzi do automatycznego podpowiadania: ***password***
4. EMS: ***10***
5. Minimalna wysokość: ***50sp***
6. Margines dolny: ***30dp***
7. Typ pola: ***hasło tekstowe***

vii. Element loginButton:
1. Szerokość: ***dopasowanie do rodzica***
2. Wysokość: ***dopasowanie do zawartości***
3. Tinta tła: ***#FF001577***

4. Wymagania dotyczące layoutu ***activity_logged_in.xml***:
Widok przedstawia zrzut ekranu:

Wymagania dla widoku:

a. Struktura drzewa komponentów przedstawiona na zrzucie ekranu:



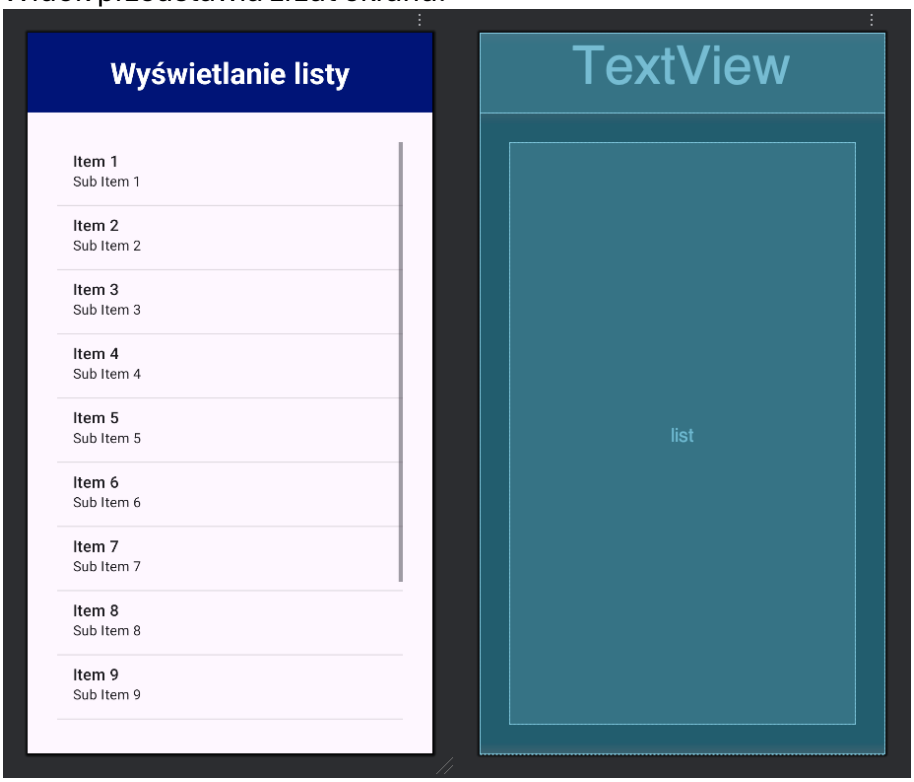
b. Wymagania dotyczące stylowania:
   i. Nagłówek z tekstem „Zabawa fragmentami”:
      1. Wysokość: ***dopasowanie do zawartości***
      2. Szerokość: ***dopasowanie do rodzica***
      3. Margines wewnętrzny: ***20dp***
      4. Ułożenie tekstu: ***do środka***
      5. Rozmiar tekstu: ***30sp***
      6. Kolor tła: ***#FF001577***
      7. Kolor tekstu: ***#FFFFFFFF***
      8. Styl testu: ***pogrubiony***
   ii. Zagnieżdżony element LinearLayout:
      1. Wysokość: ***dopasowanie do rodzica***
      2. Szerokość: ***dopasowanie do rodzica***

3. Margines wewnętrzny: ***30dp***

iii. Element changeFragmentButton:
1. Szerokość: ***dopasowanie do rodzica***
2. Wysokość: ***dopasowanie do zawartości***
3. Tinta tła: ***#FF001577***

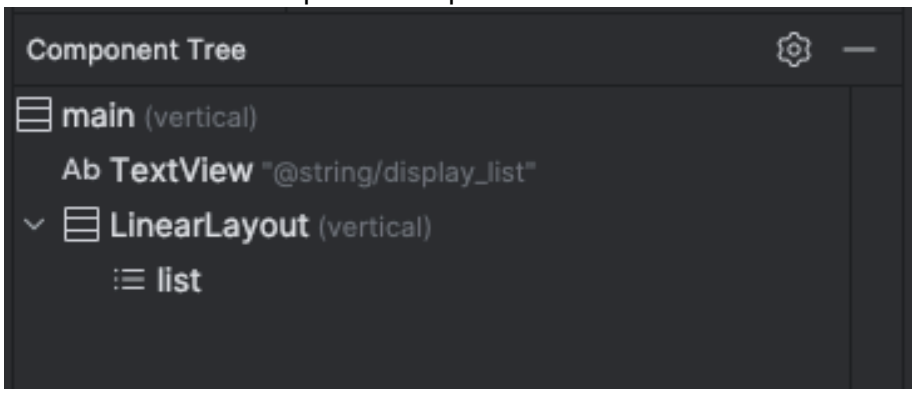
iv. Element showNotificationButton:
1. Szerokość: ***dopasowanie do rodzica***
2. Wysokość: ***dopasowanie do zawartości***
3. Tinta tła: ***#FF001577***

5. Wymagania dotyczące layoutu ***activity_notification.xml***:
Widok przedstawia zrzut ekranu:



Wymagania dla widoku:

a. Struktura drzewa komponentów przedstawiona na zrzucie ekranu:



b. Wymagania dotyczące stylowania:

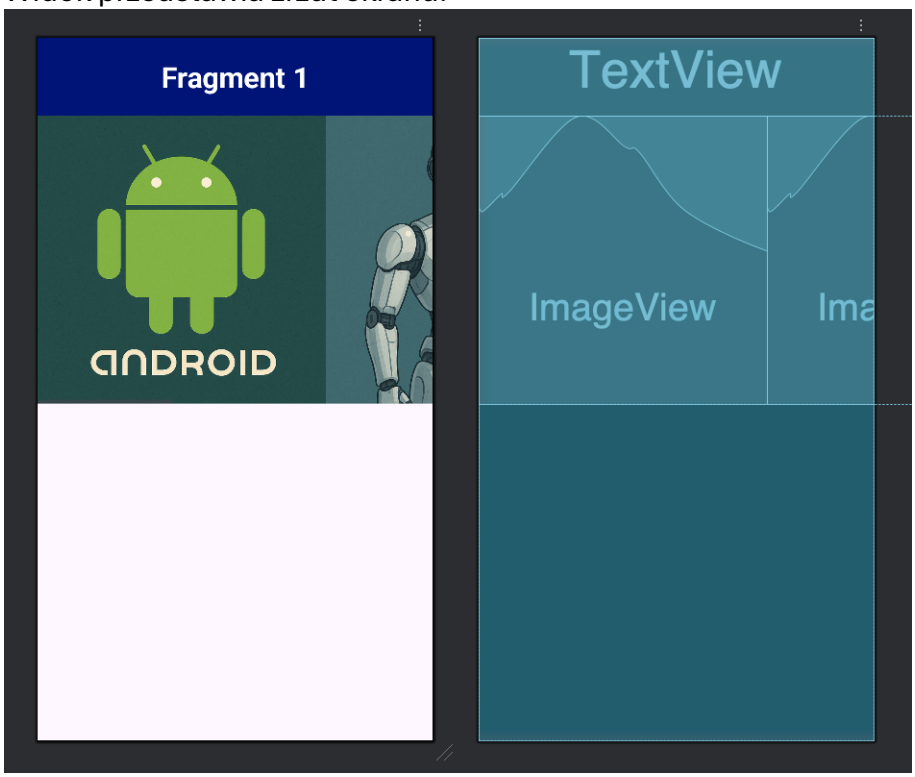
i. Nagłówek z tekstem „Wyświetlanie listy”:
1. Wysokość: ***dopasowanie do zawartości***
2. Szerokość: ***dopasowanie do rodzica***
3. Margines wewnętrzny: ***20dp***

4. Ułożenie tekstu: ***do środka***
5. Rozmiar tekstu: ***30sp***
6. Kolor tła: ***#FF001577***
7. Kolor tekstu: ***#FFFFFFFF***
8. Styl testu: ***pogrubiony***

ii. Zagnieżdżony element LinearLayout:

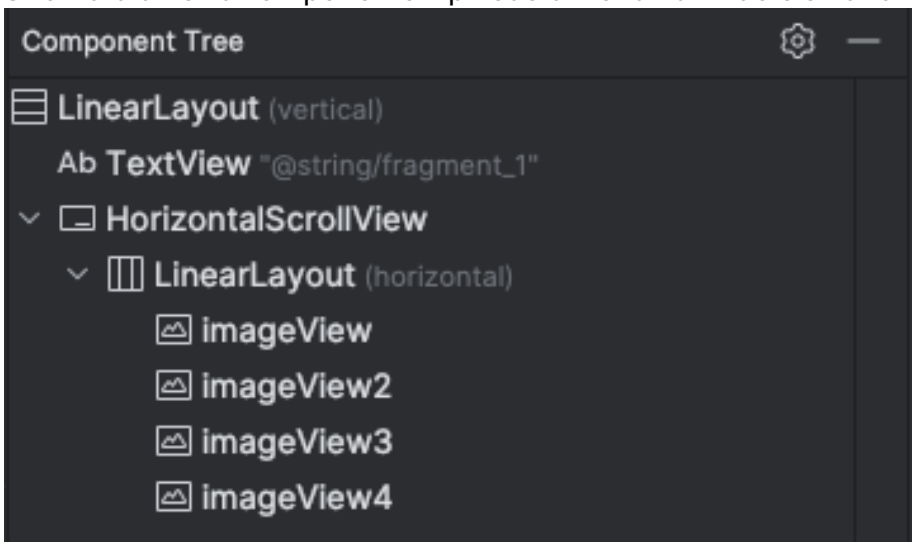
1. Wysokość: ***dopasowanie do rodzica***
2. Szerokość: ***dopasowanie do rodzica***
3. Margines wewnętrzny: ***30dp***

6. Wymagania dotyczące fragmentu ***fragment_first.xml***:
   Widok przedstawia zrzut ekranu:



Wymagania dla widoku:

a. Struktura drzewa komponentów przedstawiona na zrzucie ekranu:



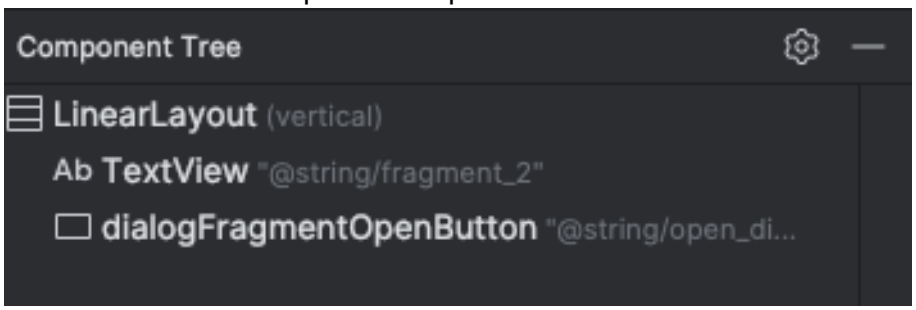
b. Wymagania dotyczące stylowania:

i. Nagłówek z tekstem „Fragment 1”:
    1. Wysokość: ***dopasowanie do zawartości***
    2. Szerokość: ***dopasowanie do rodzica***
    3. Margines wewnętrzny: ***20dp***
    4. Ułożenie tekstu: ***do środka***
    5. Rozmiar tekstu: ***30sp***
    6. Kolor tła: ***#FF001577***
    7. Kolor tekstu: ***#FFFFFFFF***
    8. Styl testu: ***pogrubiony***
ii. Element HorizontalScrollView:
    1. Szerokość: ***dopasowanie do rodzica***
    2. Wysokość: ***300dp***
iii. Elementy ImageView (4 sztuki)
    1. Szerokość: ***300dp***
    2. Wysokość: ***dopasowanie do rodzica***
    3. layout_weight: ***1***
    4. Opis zawartości: ***Android X***, gdzie ***X*** oznacza numer obrazka od 1 do 4
    5. Źródło: ***plik imgX.png***, gdzie ***X*** oznacza numer obrazka od 1 do 4. Obrazki są załączone w plikach do zadania.

7. Wymagania dotyczące fragmentu ***fragment_second.xml***:
   Widok przedstawia zrzut ekranu:

   Wymagania dla widoku:

a. Struktura drzewa komponentów przedstawiona na zrzucie ekranu:



b. Wymagania dotyczące stylowania:
   i. Nagłówek z tekstem „Fragment 2”:
      1. Wysokość: ***dopasowanie do zawartości***
      2. Szerokość: ***dopasowanie do rodzica***
      3. Margines wewnętrzny: ***20dp***
      4. Ułożenie tekstu: ***do środka***
      5. Rozmiar tekstu: ***30sp***
      6. Kolor tła: ***#FF001577***
      7. Kolor tekstu: ***#FFFFFFFF***
      8. Styl testu: ***pogrubiony***
   ii. Element dialogFragmentOpenButton:
      1. Szerokość: ***dopasowanie do rodzica***
      2. Wysokość: ***dopasowanie do zawartości***
      3. Tinta tła: ***#FF001577***

8. Wymagania dotyczące aktywności ***MainActivity***. Aktywność zawiera następujące metody:
   a. ***protected void onCreate(Bundle savedInstanceState)***:
      i. Metoda podpina widok ***activity_main.xml***
      ii. Metoda wywołuje metodę ***initDb()***
      iii. Metoda pobiera z UI przycisk ***loginButton***, a następnie przypisuje mu nasłuchiwanie zdarzenia kliknięcia. Po kliknięciu przycisku wykonywana jest następująca logika:
         1. Z UI pobierane są dane wpisane w pola ***emailInput*** oraz ***passwordInput***.
         2. Jeżeli którekolwiek z pól jest puste zostaje wyświetlony ***ToasMessage*** o treści ***„Wypełnij wszystkie pola!”***. Czas wyświetlenia Toasta to ***Toast.LENGTH_LONG***
         3. Jeżeli metoda ***checkCredentials*** zwróci fałsz zostaje wyświetlony ***ToasMessage*** o treści ***„Niepoprawne dane logowania!”***. Czas wyświetlenia Toasta to ***Toast.LENGTH_LONG***. Zawartość pola ***passwordInput*** powinna zostać wyczyszczona.
         4. Ostatecznie zostaje wyświetlony ***ToasMessage*** o treści ***„Zalogowano!”***. Czas wyświetlenia Toasta to ***Toast.LENGTH_LONG***. Zostaje wysyłana intencja uruchomienia aktywności ***LoggedInActivity***.
   b. ***private void initDb()***:
      i. Metoda tworzy bazę danych, jeżeli baza nie istnieje. Implementacja dostępu do bazy danych dowolna (API SQLite lub Room).

ii. Metoda tworzy tabelę ***users*** zawierającą dane użytkowników, jeżeli tabela nie istnieje. Tabela zawiera następujące pola:
    1. ***id*** – klucz główny, autoinkrementowany
    2. ***email*** – typ tekstowy
    3. ***password*** – typ tekstowy
iii. Jeżeli ilość wierszy w tabeli jest równa 0 do bazy wstawiane są dane następujących użytkowników:
    1. Email: admin@example.com
       hasło: admin
    2. Email: user1@example.com
       hasło: user1
    3. Email: user2@example.com
       hasło: user2
    4. Email: user3@example.com
       hasło: user3

c. ***private boolean checkCredentials(String email, String password)***:
    i. Metoda pobiera użytkownika na podstawie adresu e-mail (argument email).
    ii. Jeżeli użytkownik nie został znaleziony w bazie danych, to zwraca fałsz.
    iii. Jeżeli użytkownik został znaleziony metoda sprawdza, czy hasło zapisane w bazie danych jest zgodne z argumentem ***password*** i zwraca odpowiednią wartość logiczną.

9. Wymagania dotyczące aktywności ***LoggedInActivity***. Aktywność zawiera następujące pola i metody:
    a. Pole ***CHANNEL_ID*** o wartości „***2137***”, typ String, stała
    b. Pole ***CHANNEL_NAME*** o wartości „***Zadanie podsumowujące***”, typ String, stała
    c. Pole ***activeFragment***, typ String
    d. Pole ***fragmentManager***, typ FragmentManager
    e. ***protected void onCreate(Bundle savedInstanceState)***:
        i. Metoda podpina widok ***activity_logged_in.xml***
        ii. Metoda przypisuje do pola ***fragmentManager*** wartość wspieranego fragment managera.
        iii. Metoda umieszcza w elemencie UI ***fragmentHolder*** zawartość fragmentu ***FirstFragment***.
        iv. Metoda ustawia nasłuchiwanie zdarzenia kliknięcia na przycisk ***changeFragmentButton*** jako wywołanie metody ***changeFragment***.
        v. Metoda ustawia nasłuchiwanie zdarzenia kliknięcia na przycisk ***showNotificationButton*** jako wywołanie metody ***sendNotification***.
    f. ***private void changeFragment()***:
        i. Metoda sprawdza czy wartość pola ***activeFragment*** jest równa „first”. Jeżeli tak, to zmienia wartość pola na „second” oraz podmienia aktywny fragment na ***SecondFragment***

ii. W przeciwnym wypadku zmienia wartość pola na „first” oraz podmeinia aktywny fragment na ***FristFragment***

g. ***private void sendNotification()***:
   i. Metoda tworzy opóźnioną intencję otwarcia aktywności ***NotificationActivity***.
   ii. Metoda tworzy kanał powiadomień o ID ***CHANNEL_ID*** oraz nazwie ***CHANNEL_NAME***.
   iii. Metoda wysyła powiadomienie o następujących właściwościach:
      1. Tytuł: ***Powiadomienie***
      2. Treść: ***Wiadomość powiadomienia***
      3. Akcja: ***Utworzona wcześniej opóźniona intencja***

h. ***public void openDialog()***:
   i. Metoda tworzy nową instancję fragmentu ***AppDialogFragment***.
   ii. Metoda ustawia możliwość zamknięcia okna dialogowego.
   iii. Metoda wywołuje pokazanie okna dialogowego.

10. Wymagania dotyczące fragmentu ***FirstFragment***. Fragment zawiera następujące metody:
    a. ***public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)***:
       i. Metoda podpina widok ***fragment_first.xml***
11. Wymagania dotyczące fragmentu ***SecondFragment***. Fragment zawiera następujące metody:
    a. ***public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)***:
       i. Metoda podpina widok ***fragment_second.xml***
       ii. Metoda znajduje w UI element ***dialogFragmentOpenButton*** i ustawia nasłuchiwanie zdarzenia kliknięcia na przycisk jako wywołanie metody ***openDialog*** z aktywności wyświetlającej fragment (***LoggedInActivity***).
12. Wymagania dotyczące aktywności ***NotificationActivity***. Aktywność zawiera następujące pola i metody:
    a. Pole ***LIST_ITEMS***, stała tablica String o zawartości:
       i. „Programowanie obiektowe”
       ii. „Programowanie Aplikacji Mobilnych”
       iii. „Programowanie Aplikacji Internetowych”
       iv. „Programowanie Aplikacji Webowych”
       v. „Programowanie Aplikacji Desktopowych”
    b. ***protected void onCreate(Bundle savedInstanceState)***:
       i. Metoda podpina widok ***activity_notification.xml***
       ii. Metoda tworzy ArrayAdapter zawierający elementy z pola ***LIST_ITEMS*** wyświetlane jako ***android.R.layout.simple_list_item_1***
       iii. Metoda przypisuje adapter do elementu ***list***