REPORT
LAB 5: XYLOPHONE

Student name: Pham Quoc Huy

Student ID: 21IT413

Student email: huypq.21it@vku.udn.vn

1. **Introduction**

- The purpose of this lab report is to document the development of a simple xylophone mobile application using Flutter. The xylophone app aims to provide a basic soundplaying functionality, where each button on the screen corresponds to a different note of the xylophone.

  The project explores the development of a cross-platform app with a user interface and sound interaction

2. **Objectives**

- To learn how to use Flutter to build a simple interactive mobile app.
- To gain hands-on experience with cross-platform development using the Flutter framework.
- To implement sound-playing functionality using the audioplayers package.

3. **Methodology**

The methodology followed in this lab involved several stages:

1. **Setting Up the Environment:** The Flutter framework was used for crossplatform development. Android Studio was utilized for coding and running the app.
2. **Creating the User Interface:** The app features a simple vertical layout containing seven buttons, each corresponding to a note of the xylophone. These buttons were implemented using Flutter widgets such as Expanded and TextButton.
3. **Implementing Sound Functionality**: The audioplayers package was integrated to handle sound playback. Each button is connected to an audio file representing a note, and these files are triggered when the user taps the buttons.

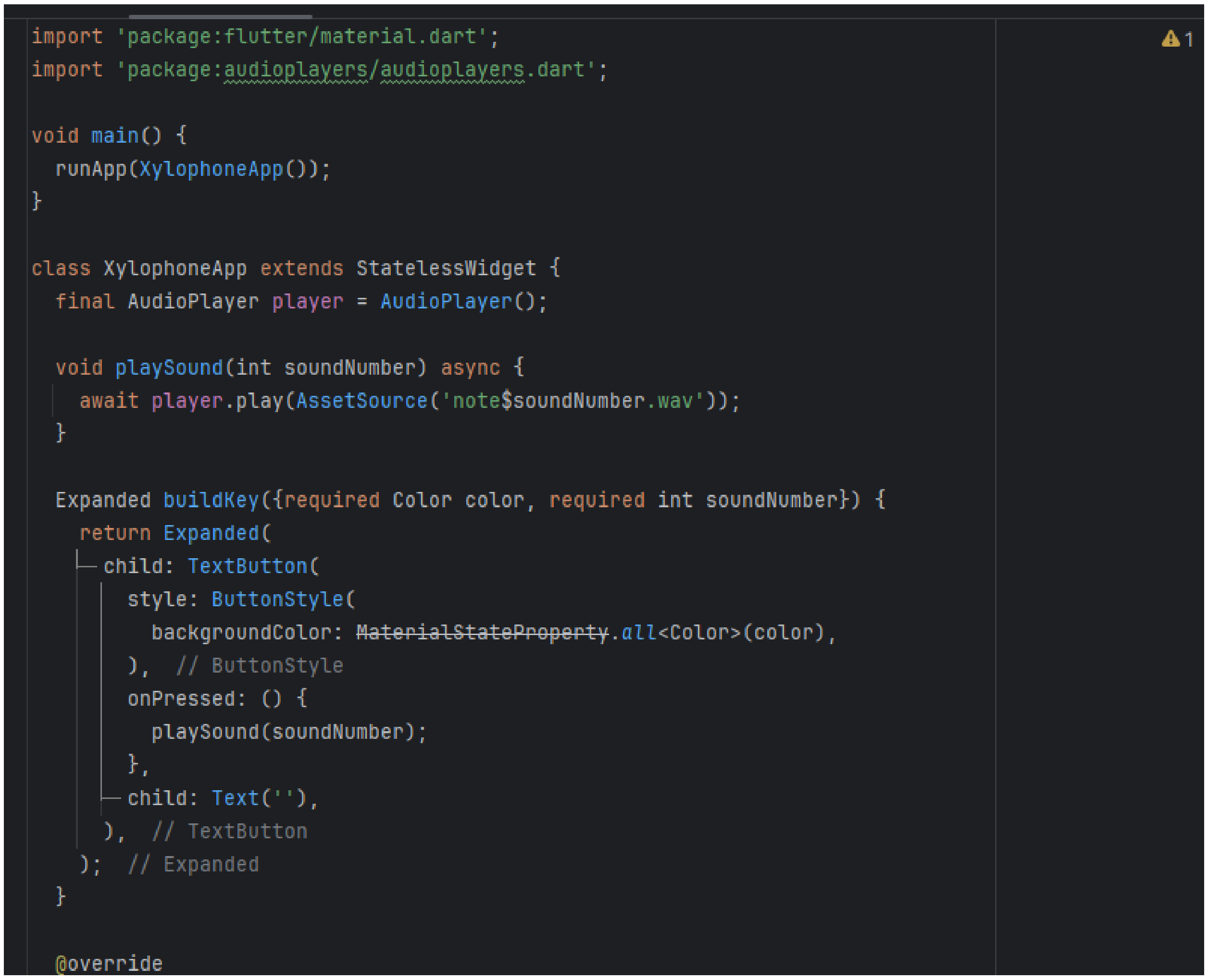
4. **Testing:** The app was tested on an Android device to ensure that the sound files played correctly

4. Results

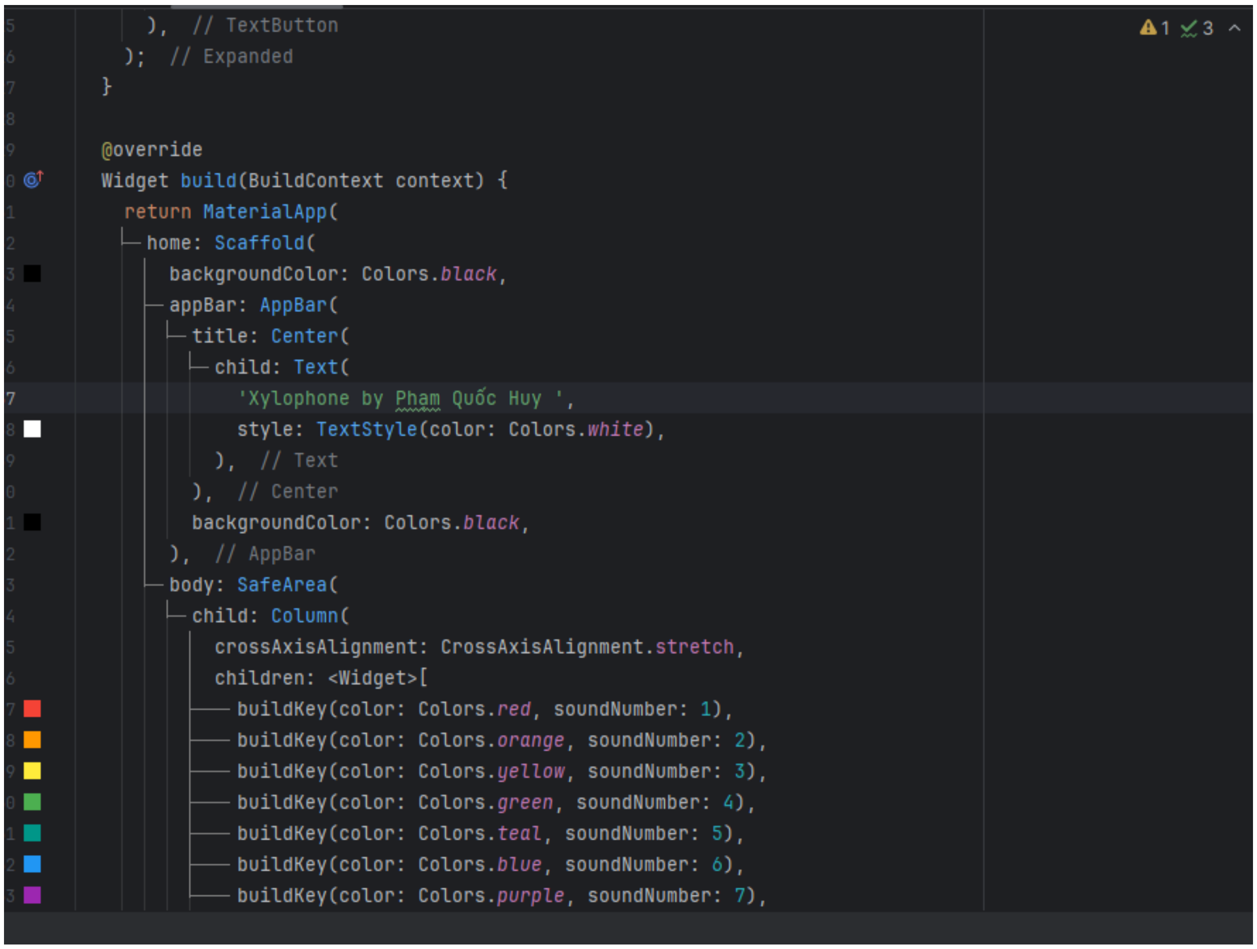
- The final app is a simple xylophone instrument that successfully plays different notes when each button is tapped. The user interface is responsive, and sound playback occurs without noticeable delay

Screenshot of the App:

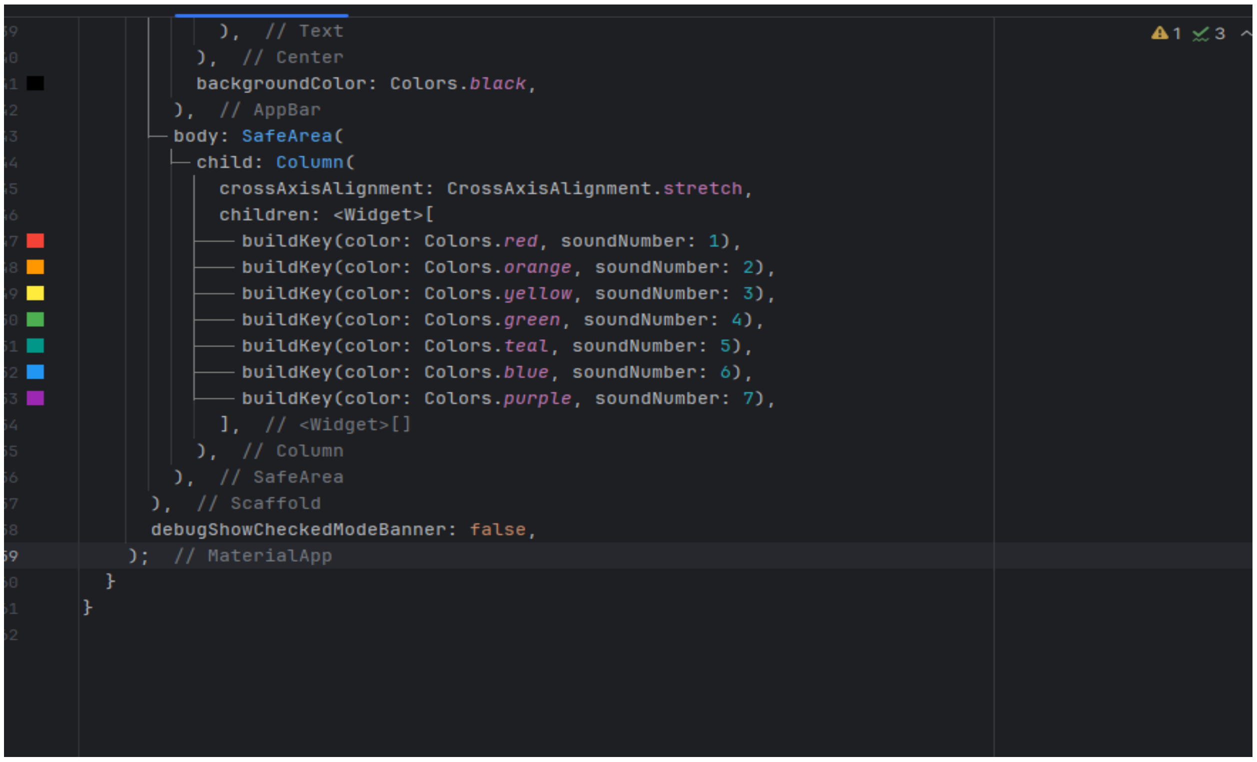
- Screenshots of the app showing the UI and buttons layout
    - **Main.dart**



```dart
import 'package:flutter/material.dart';
import 'package:audioplayers/audioplayers.dart';

void main() {
  runApp(XylophoneApp());
}

class XylophoneApp extends StatelessWidget {
  final AudioPlayer player = AudioPlayer();

  void playSound(int soundNumber) async {
    await player.play(AssetSource('note$soundNumber.wav'));
  }

  Expanded buildKey({required Color color, required int soundNumber}) {
    return Expanded(
      child: TextButton(
        style: ButtonStyle(
          backgroundColor: ~~MaterialStateProperty~~.all<Color>(color),
        ),  // ButtonStyle
        onPressed: () {
          playSound(soundNumber);
        },
        child: Text(''),
      ),  // TextButton
    );  // Expanded
  }

  @override
```

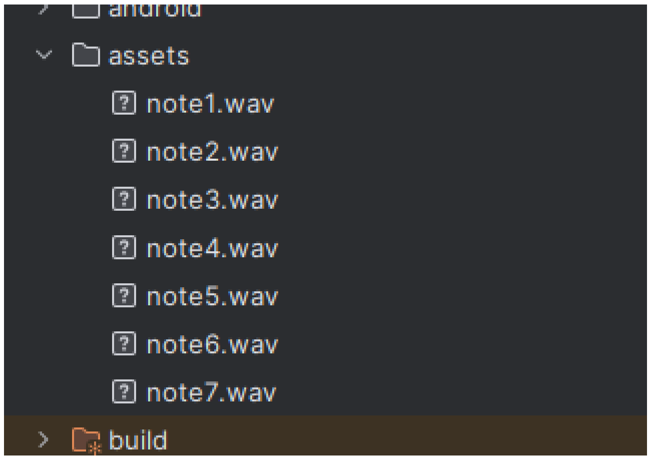


```
      ),  // TextButton
    );  // Expanded
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        backgroundColor: Colors.black,
        appBar: AppBar(
          title: Center(
            child: Text(
              'Xylophone by Pham Quốc Huy ',
              style: TextStyle(color: Colors.white),
            ),  // Text
          ),  // Center
          backgroundColor: Colors.black,
        ),  // AppBar
        body: SafeArea(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.stretch,
            children: <Widget>[
              buildKey(color: Colors.red, soundNumber: 1),
              buildKey(color: Colors.orange, soundNumber: 2),
              buildKey(color: Colors.yellow, soundNumber: 3),
              buildKey(color: Colors.green, soundNumber: 4),
              buildKey(color: Colors.teal, soundNumber: 5),
              buildKey(color: Colors.blue, soundNumber: 6),
              buildKey(color: Colors.purple, soundNumber: 7),
```



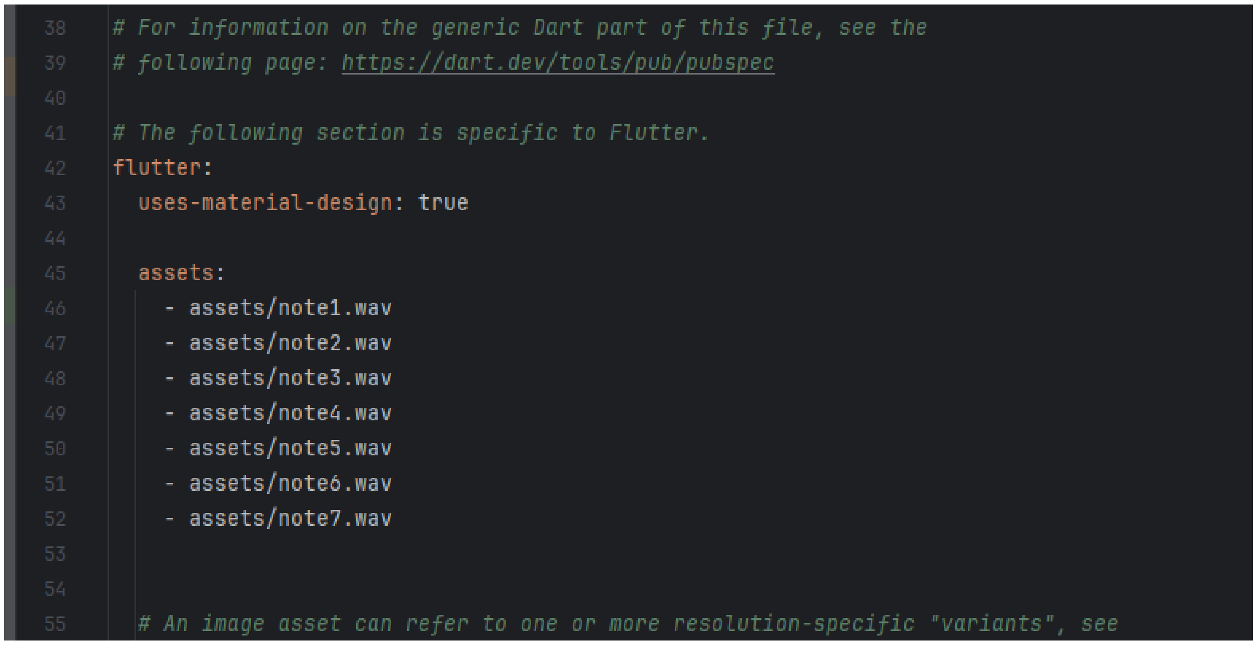
```
            ),  // Text
          ),  // Center
          backgroundColor: Colors.black,
        ),  // AppBar
        body: SafeArea(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.stretch,
            children: <Widget>[
              buildKey(color: Colors.red, soundNumber: 1),
              buildKey(color: Colors.orange, soundNumber: 2),
              buildKey(color: Colors.yellow, soundNumber: 3),
              buildKey(color: Colors.green, soundNumber: 4),
              buildKey(color: Colors.teal, soundNumber: 5),
              buildKey(color: Colors.blue, soundNumber: 6),
              buildKey(color: Colors.purple, soundNumber: 7),
            ],  // <Widget>[]
          ),  // Column
        ),  // SafeArea
      ),  // Scaffold
      debugShowCheckedModeBanner: false,
    );  // MaterialApp
  }
}
```

- **Directory Structure:**



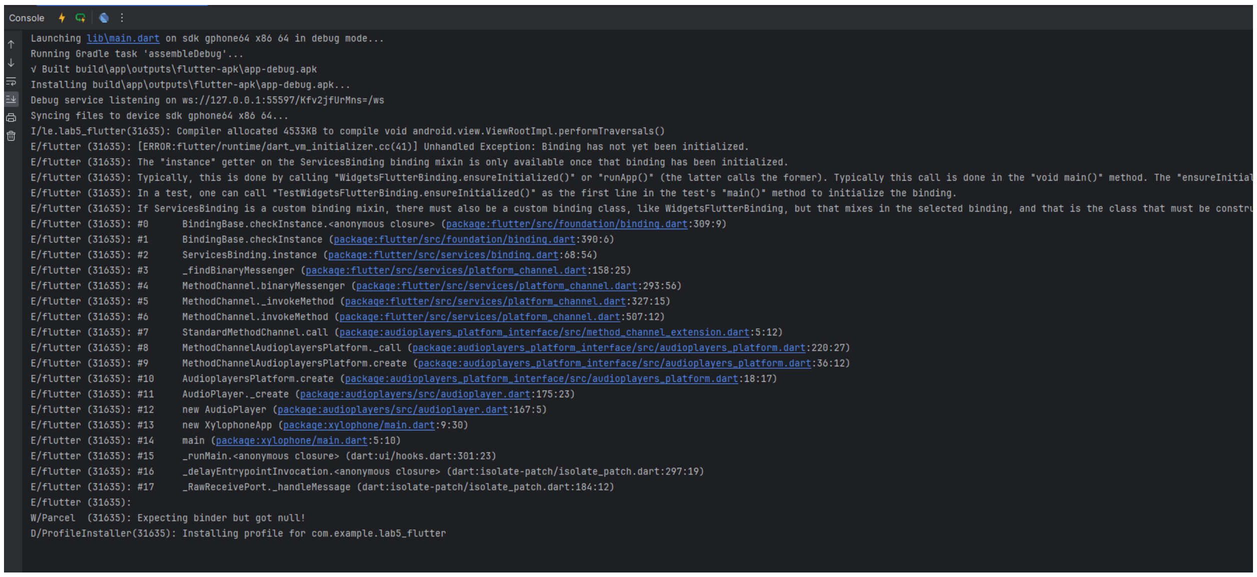
```
> android
v assets
    note1.wav
    note2.wav
    note3.wav
    note4.wav
    note5.wav
    note6.wav
    note7.wav
> build
```

- **Pubspec.yaml**



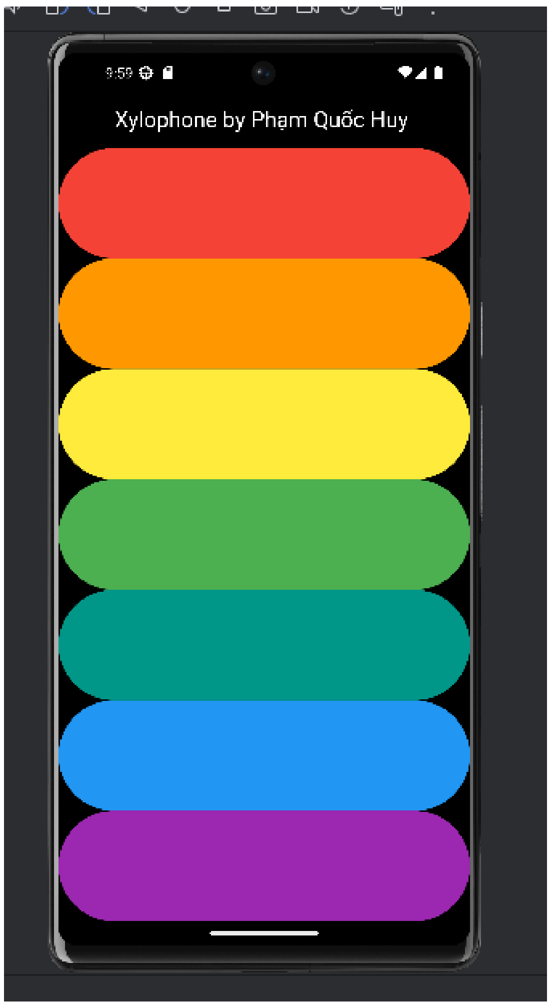
```yaml
# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec

# The following section is specific to Flutter.
flutter:
  uses-material-design: true

  assets:
    - assets/note1.wav
    - assets/note2.wav
    - assets/note3.wav
    - assets/note4.wav
    - assets/note5.wav
    - assets/note6.wav
    - assets/note7.wav


  # An image asset can refer to one or more resolution-specific "variants", see
```

- **Console:**



```
Launching lib\main.dart on sdk gphone64 x86 64 in debug mode...
Running Gradle task 'assembleDebug'...
√ Built build\app\outputs\flutter-apk\app-debug.apk
Installing build\app\outputs\flutter-apk\app-debug.apk...
Debug service listening on ws://127.0.0.1:55597/Kfv2jfUrMns=/ws
Syncing files to device sdk gphone64 x86 64...
I/le.lab5_flutter(31635): Compiler allocated 4533KB to compile void android.view.ViewRootImpl.performTraversals()
E/flutter (31635): [ERROR:flutter/runtime/dart_vm_initializer.cc(41)] Unhandled Exception: Binding has not yet been initialized.
E/flutter (31635): The "instance" getter on the ServicesBinding binding mixin is only available once that binding has been initialized.
E/flutter (31635): Typically, this is done by calling "WidgetsFlutterBinding.ensureInitialized()" or "runApp()" (the latter calls the former). Typically this call is done in the "void main()" method. The "ensureInitial
E/flutter (31635): In a test, one can call "TestWidgetsFlutterBinding.ensureInitialized()" as the first line in the test's "main()" method to initialize the binding.
E/flutter (31635): If ServicesBinding is a custom binding mixin, there must also be a custom binding class, like WidgetsFlutterBinding, but that mixes in the selected binding, and that is the class that must be constru
E/flutter (31635): #0      BindingBase.checkInstance.<anonymous closure> (package:flutter/src/foundation/binding.dart:309:9)
E/flutter (31635): #1      BindingBase.checkInstance (package:flutter/src/foundation/binding.dart:390:6)
E/flutter (31635): #2      ServicesBinding.instance (package:flutter/src/services/binding.dart:68:54)
E/flutter (31635): #3      _findBinaryMessenger (package:flutter/src/services/platform_channel.dart:158:25)
E/flutter (31635): #4      MethodChannel.binaryMessenger (package:flutter/src/services/platform_channel.dart:293:56)
E/flutter (31635): #5      MethodChannel._invokeMethod (package:flutter/src/services/platform_channel.dart:327:15)
E/flutter (31635): #6      MethodChannel.invokeMethod (package:flutter/src/services/platform_channel.dart:507:12)
E/flutter (31635): #7      StandardMethodChannel.call (package:audioplayers_platform_interface/src/method_channel_extension.dart:5:12)
E/flutter (31635): #8      MethodChannelAudioplayersPlatform._call (package:audioplayers_platform_interface/src/audioplayers_platform.dart:220:27)
E/flutter (31635): #9      MethodChannelAudioplayersPlatform.create (package:audioplayers_platform_interface/src/audioplayers_platform.dart:36:12)
E/flutter (31635): #10     AudioplayersPlatform.create (package:audioplayers_platform_interface/src/audioplayers_platform.dart:18:17)
E/flutter (31635): #11     AudioPlayer._create (package:audioplayers/src/audioplayer.dart:175:23)
E/flutter (31635): #12     new AudioPlayer (package:audioplayers/src/audioplayer.dart:167:5)
E/flutter (31635): #13     new XylophoneApp (package:xylophone/main.dart:9:30)
E/flutter (31635): #14     main (package:xylophone/main.dart:5:10)
E/flutter (31635): #15     _runMain.<anonymous closure> (dart:ui/hooks.dart:301:23)
E/flutter (31635): #16     _delayEntrypointInvocation.<anonymous closure> (dart:isolate-patch/isolate_patch.dart:297:19)
E/flutter (31635): #17     _RawReceivePort._handleMessage (dart:isolate-patch/isolate_patch.dart:184:12)
E/flutter (31635):
W/Parcel  (31635): Expecting binder but got null!
D/ProfileInstaller(31635): Installing profile for com.example.lab5_flutter
```

- Screenshot of the app running on the Android device:



5. Discussion

- The results obtained in this lab were successful, with the app functioning as intended. The sound files played correctly, and the user interface was simple yet effective for its purpose.

Strengths of Cross-Platform Development:

- The ability to write code once and deploy on multiple platforms is a significant advantage of using Flutter.
- Flutter provides a rich set of pre-built widgets, making UI design faster and more consistent

Weaknesses:

- Cross-platform development may not offer the same level of optimization as native development for platform-specific features like performance-intensive tasks.
- Some issues with package compatibility across platforms may arise during development.

6. Conclusion

- The xylophone app demonstrated the effectiveness of Flutter for building simple mobile apps with sound functionality. The app met the objectives by playing musical notes upon interaction and using cross-platform tools efficiently.
- For future work, it is recommended to explore more advanced sound features, such as volume control and sound mixing, and to extend the app's functionality with additional instruments