Smart Pill Dispenser

PROJECT REPORT

WINTER SEM 2022-23

ECS1002

SUBMITTED BY

ADITYA PATIL – 20BCE7013

DHYAN –20BCE7648

IVAN CHRISTOPHER – 20BCI7097

JEEVANTIKA GURUNG – 20BCR7087

KHAJAN – 20BES7003

PRIYAM – 20BES7012

GUIDED BY

DR. AMBUJ SHARMA

Index

Smart Pill Dispenser

Aditya Patil, Dhyan, Ivan Christopher, Jeevantika Gurung, Khajan, Priyam

Abstract: This report describes the design and development of a smart pill dispenser for people with autism, the elderly, and those with disabilities to manage their medication. The device features a mobile app, cloud-based real-time updating, and pill consumption reminders. The aim is to reduce the risk of missed or incorrect medication doses and improve the quality of life for these vulnerable populations. The smart pill dispenser is user-friendly, compact, and accessible to individuals of all ages and abilities.

I. Introduction

Medication management is a crucial aspect of maintaining good health, especially for induvial with autism, the elderly, and people with disabilities. These populations often require medication for various medical conditions, and any missed or incorrect doses can lead to serious health complications [2]. However, managing medication can be challenging, especially for those with cognitive or physical impairments.

The aim of this project is to design and develop a mobile smart pill box to assist individuals with autism, the elderly, and people with disabilities in managing their medication and overall health. The smart pill box is designed to be user-friendly, compact, and equipped with advanced features to make it accessible to people of all ages and abilities.

II. Problem Definition

The problem this project aims to solve is the difficulty faced by individuals who are visually impaired, hearing impaired, or have autism in managing their medication. These populations often require assistance from a caregiver to take their medication, which can be inconvenient and may compromise their privacy. Missed or incorrect doses of medication can have serious consequences for these populations, including worsening of their health condition or potential hospitalization. The smart pill dispenser provides a solution to this problem by allowing individuals to manage their medication independently and effectively. The device is easy to operate and can be customized to accommodate individual pill configurations, making it accessible to people of all ages and abilities.

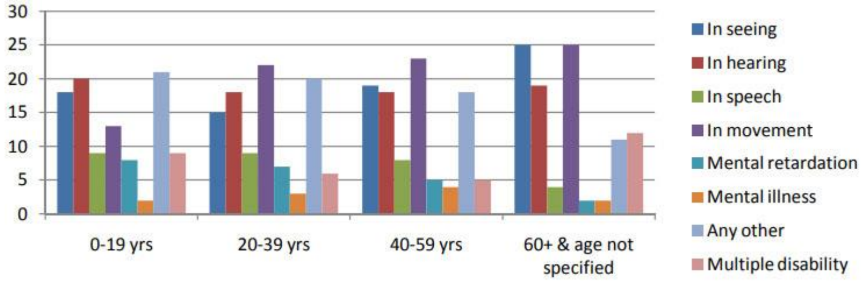


Figure 1- Statistic on Disabled Population in India [1]

III. OBJECTIVES

- To **design and develop** a smart pill dispenser that can be easily operated by individuals with visual, hearing, or cognitive impairments.
- To **create** an *Android application* that can be used to input and store pill configuration details and establish a connection with AWS S3 cloud storage.
- To **develop** a **backend infrastructure** using AWS services such as Lambda, SNS, and S3 to store and manage pill configuration data.
- To **integrate** the smart pill dispenser with the Raspberry Pi to enable timely and accurate *dispensing of medication*.
- To **create** an alarm system and LED flash mechanism that will alert users to the appropriate dosage and timing of their medication. [4]
- To **incorporate** an *IR sensor* to identify the container and dispense the **correct amount of medication**.
- To test the device and app for accuracy, usability, and reliability in dispensing medication according to individualized pill configurations.
- To **evaluate** the user experience and satisfaction with the smart pill dispenser and make *necessary adjustments based on feedback*.
- To provide an **affordable** and **accessible** solution for medication management that improves the quality of life for individuals with *visual, hearing, or cognitive impairments*.

IV. METHODOLOGY

The procedure to bring about the pill dispenser followed a creation pipeline as follows:

1) **Research** existing smart pill dispenser technologies and available resources for developing the project.
2) Conduct a **needs assessment** to identify the specific requirements of individuals with visual, hearing, or cognitive impairments in managing their medication.
3) **Design a prototype** of the smart pill dispenser and test its functionality with a Raspberry Pi, IR sensor, SG motors, and display. [3]
4) Develop an **Android application** that can be used to input and store pill configuration details and establish a connection with AWS S3 cloud storage.

A. Mobile App (Android Studio)

SMART PILL DISPENSER(UI)

- Android OS: 11.0 (R) - API 30
- Development Platform: Android Studio
- Minimum SDK: 32
- RAM Required (Min): 4
- Internet Connectivity

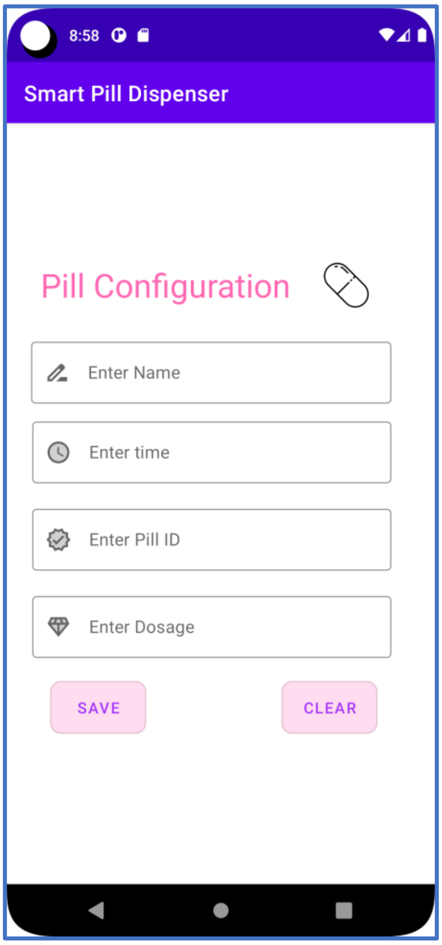


Figure 2- Screenshot of app

- Create a backend infrastructure using AWS services such as Lambda, SNS, and S3 to store and manage pill configuration data.

B. Cloud Service (AWS)

1. *Simple Storage Service (S3)*

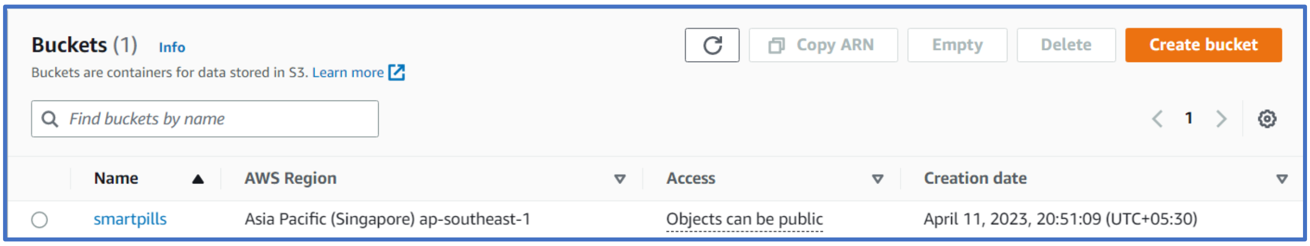


Figure 3 - Bucket created to store the calls

2. *Lambda Function*

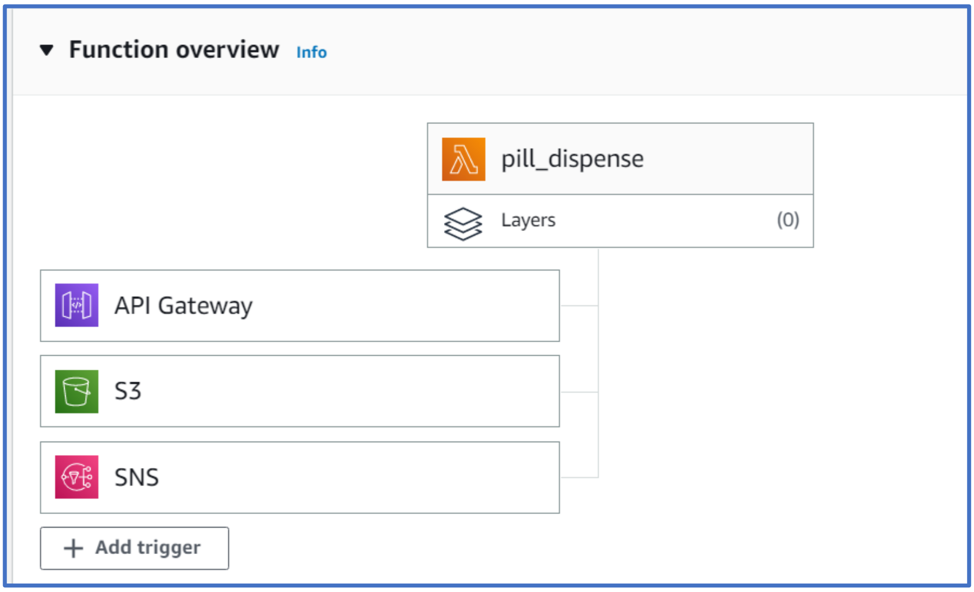


Figure 4-Lambda Function

3. *Simple Notification Service (SNS)*

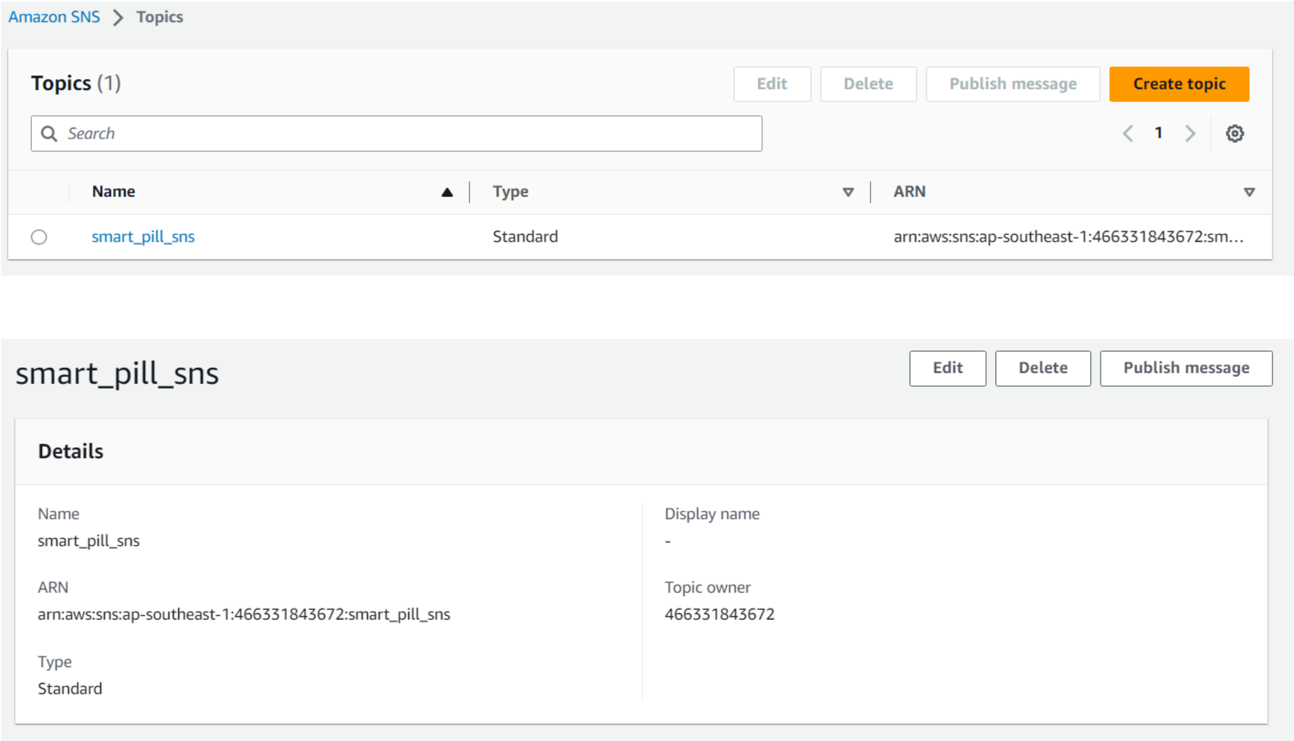


Figure 5 - Screenshot of Notification Service

We integrated the smart pill dispenser with the Raspberry Pi and established communication between the dispenser and the Android app using Wi-Fi connectivity.

An alarm system and LED flash mechanism were developed to alert users to the appropriate dosage and timing of their medication.

An IR sensor was also incorporated to identify the container and dispense the correct amount of medication.

The device and app were then tested for accuracy, usability, and reliability in dispensing medication according to individualized pill configurations.

C. **Smart Pill Box**

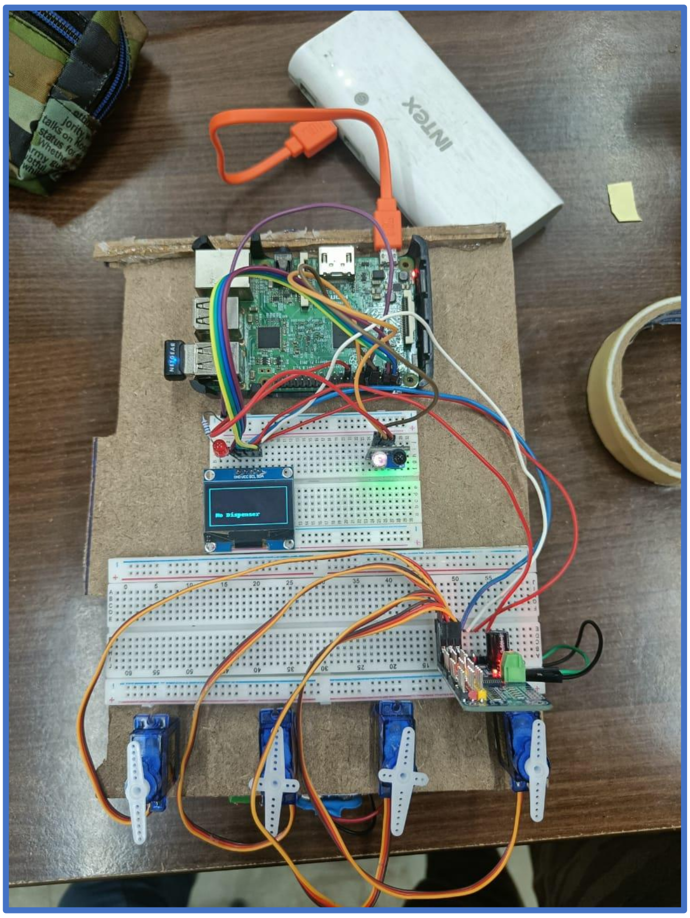


Figure 6 - Inner Electronics

We evaluated the user experience and satisfaction with the smart pill dispenser through user testing and feedback. Based on this feedback, we made necessary adjustments to the device and app, and conducted repeat testing as needed.

V. RESULTS AND DISCUSSION

The smart pill dispenser was successfully designed and developed to assist individuals with visual, hearing, or cognitive impairments in managing their medication. The dispenser was able to hold up to 5 types of pills and could be easily configured through an Android app, allowing users to input pill configuration details such as name, pill ID, dosage, and time. These details were then stored in AWS S3 cloud storage, which triggered a Lambda function to send reminders to the user's registered email address via AWS SNS service.

The dispenser was powered by a Raspberry Pi and equipped with an IR sensor, SG motors, and display. The Raspberry Pi received the pill configuration details, and when the specified time was reached, an alarm was triggered to alert visually impaired users, and an LED was flashed for users with hearing impairments. The dispenser accurately dispensed the appropriate number of pills into a container identified by an IR sensor, ensuring that users received the correct dosage at the appropriate time.

Testing showed that the device and app were accurate, reliable, and easy to use. User testing and feedback helped to identify areas for improvement, including enhancing the alarm system and LED flash mechanism and making the Android app more user-friendly.

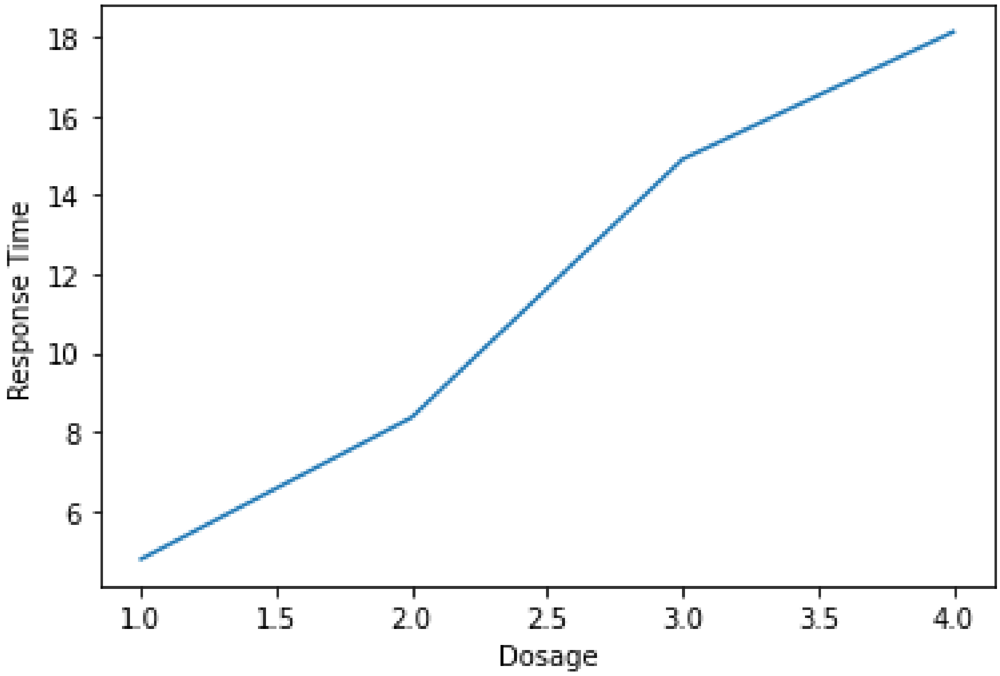
TESTING

The Pill Dispenser was run **100 times** every **2 minutes**. The dosages were random from 1-5.

	Timestamp	Response Time	Patient ID	Medication Name	Dosage
0	2023-04-19 00:00:00	14.370870	Test#-1	Med1	2
1	2023-04-19 00:02:00	20.898054	Test#-2	Med3	3
2	2023-04-19 00:04:00	17.114975	Test#-3	Med3	2
3	2023-04-19 00:06:00	7.078120	Test#-4	Med1	4
4	2023-04-19 00:08:00	8.891657	Test#-5	Med1	2
...	...	...	...	...	...
95	2023-04-19 03:10:00	5.805536	Test#-96	Med3	1
96	2023-04-19 03:12:00	31.155731	Test#-97	Med1	4
97	2023-04-19 03:14:00	0.146871	Test#-98	Med3	1
98	2023-04-19 03:16:00	1.284861	Test#-99	Med1	2
99	2023-04-19 03:18:00	9.279990	Test#-100	Med1	2

Figure 7 - database of test data

The response time was recorded, and we inferred the below results:



Dosage	Response Time
1	4.775777
2	8.361624
3	14.896721
4	18.121147

There was almost a linear correlation between dosage and response time

```
df["Response Time"].mean()
```

```
12.344016393397292
```

```
df["Dosage"].mean()
```

```
2.67
```

The mean response time for the testing was 12 seconds, with the mean dosage being 2.67

Dosage	Timestamp	Response Time	Patient ID	Medication Name
1	2023-04-19 03:14:00	10.100663	Test#-98	Med3
2	2023-04-19 03:18:00	25.945859	Test#-99	Med3
3	2023-04-19 03:08:00	39.895454	Test#-95	Med3
4	2023-04-19 03:12:00	40.632671	Test#-97	Med3

The worst-case scenario time for each of the dosages were as above. The reliability of the servo-motors, various physical elements come into play when such executions are made and affect the result such as friction, make of the servo motors, etc.

There were **no** cases of incorrect dosages, as the servo systems functioned perfectly.

Overall, the smart pill dispenser was able to provide an affordable and accessible solution for medication management, improving the quality of life for individuals with visual, hearing, or cognitive impairments. The successful development of this technology demonstrates the potential for further innovation in assistive technologies to enhance the lives of people with disabilities.

VI. CONCLUSION AND FUTURE SCOPE

Conclusion:
In conclusion, the development of the smart pill dispenser represents a significant contribution to assistive technology for individuals with visual, hearing, or cognitive impairments. The device provides a simple and effective solution for managing medication and can significantly reduce the risk of missed or incorrect doses.
The Android app and cloud storage system enable easy customization and update of medication details, while the Raspberry Pi-based dispenser accurately dispenses the appropriate number of pills at the right time. This innovative solution has the potential to improve the quality of life for many people and offers a practical solution to a significant problem.

Future Scope:
While this project has been successful in developing a working prototype, there is still potential for further enhancements and improvements. Here are some future scope possibilities:

- Integration with wearable devices: The smart pill dispenser could be integrated with wearable devices to provide real-time monitoring of vital signs such as heart rate and blood pressure. This integration could help identify patterns and ensure timely medication administration.
- Expansion of functionality: The device could be expanded to include other features such as temperature and humidity sensors to ensure proper storage conditions for medication.
- Integration with other healthcare providers: The device could be linked to the healthcare provider's electronic medical record system, ensuring timely medication adherence monitoring, and timely communication with healthcare providers.
- Integration with voice assistants: The smart pill dispenser could be integrated with voice assistants, making it more accessible to individuals with cognitive and visual impairments.

VII. REFERENCES

[1] https://wecapable.com/disabled-population-india-data/

[2] https://smartech.gatech.edu/bitstream/handle/1853/45009/MedicationDeliveryTechReport.pdf?sequence=1&isAllowed=y

[3] https://rspsciencehub.com/pdf_1405_1b9b149b2d622ace45d58a7a466f7783.html

[4] https://www.researchgate.net/profile/Haifeng-Wang-25/publication/319688945_Drug_Dispenser_Replenishment_Optimization_via_Mixed_Integer_Programming_in_Central_Fill_Pharmacy_Systems/links/59b98fad458515bb9c48a302/Drug-Dispenser-Replenishment-Optimization-via-Mixed-Integer-Programming-in-Central-Fill-Pharmacy-Systems.pdf

VIII. CODES

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/andr
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/name1"
        android:layout_width="330dp"
        android:layout_height="62dp"
        android:layout_marginTop="195dp"
        android:layout_marginBottom="20dp"
        android:layout_marginEnd="40dp"
        android:hint="Enter Name "
        app:boxStrokeColor="#ff69b4"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintEnd_toEndOf="parent"

        app:startIconDrawable="@drawable/baseline_drive_file_rename_outline_24"
        style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox">

        <!--this is the actual edit text which takes the input-->
        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/name"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"

        android:label="Smart Pill Dispenser"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

```groovy
android {
    namespace 'com.example.myapplication'
    compileSdk 33

    defaultConfig {
        applicationId "com.example.myapplication"
        minSdk 29
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8

    }
```