

Project Report
On
**Software for Preventive Maintenance
Reminder of
'A' vehicles and 'B' vehicles**

By
**Ayush Pal (2020B1A41933G)
Krisha Vinay Sanghvi (2020PSA71724G)
Gurle Vaishnavi (2020PSA70007P)
Manas Kumar Khandwal (2020A4PS2272H)**

At
ARMY BASE WORKSHOP, PITORAGARH
A Practice School-1 Station
of
BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI



BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Project Report
On
Software for Preventive Maintenance
Reminder of
‘A’ vehicles and ‘B’ vehicles

Assigned to

Name	I.D.	Discipline(s)/of the student(s)
Ayush Pal	2020B1A41933G	M.Sc. Biological Sciences + B.E Mech
Krishna Vinay Sanghvi	2020A7PS1724G	B.E Computer Science
Gurle Vaishnavi	2020A7PS0007P	B.E Computer Science
Manas Kumar Khandwal	2020A4PS2272H	B.E Mech

At
ARMY BASE WORKSHOP, PITORAGARH
A Practice School-1 Station
of
BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

ACKNOWLEDGMENTS

We want to express our sincerest gratitude to Sir Antariksh Khanna, our instructor in charge, for his guidance and for explaining the project to our team in a way that made it simple to get started and complete it on schedule. He offered us priceless counsel and supported us during trying times. And we owe Sir Asrarul Haque a huge debt of gratitude for allowing us the chance to work on this project, which has taught us invaluable lessons regarding software development for preventive maintenance reminders of "A" and "B" vehicles. His drive and assistance made a significant difference in the project's smooth development. We would also like to thank all those who have given us unwavering support, whether it be sharing with us their knowledge or just supporting and encouraging us. We are overwhelmed with gratitude and humility for everyone who has assisted us in giving a shape to our ideas and elevating them into something more substantial. Lastly, we would like to thank the Practice School Division for their hard work, communication and commitment to work.

BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
RAJASTHAN

Practice School Division

Station: Army Base Workshop, Pithoragarh

Centre: Pithoragarh

Duration: 7 Weeks

Date of Start: 30th May, 2022

Date of Submission: 15th July, 2022

Title of the Project: Software for Preventive Maintenance Reminder of 'A' vehicles and 'B' vehicles

Students Details:

Name	I.D.	Discipline(s)/of the student(s)
Ayush Pal	2020B1A41933G	M.Sc. Biological Sciences + B.E Mech
Krishna Vinay Sanghvi	2020A7PS1724G	B.E Computer Science
Gurle Vaishnavi	2020A7PS0007P	B.E Computer Science
Manas Kumar Khandwal	2020A4PS2272H	B.E Mech

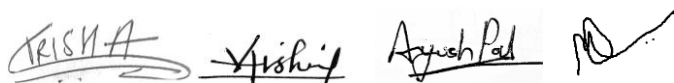
Name(s) and designation(s) of the expert(s): Nk Shobhit Kumar and Major Ravinder Jaral

Name(s) of the PS Faculty: Dr. Arnab Hazra

Key Words: Preventive Maintenance Reminder, Software Development, 'A' vehicles and 'B' vehicles

Project Areas: Software Development

Abstract: We started the project by understanding the client requirements of the maintenance reminder application we have been tasked with. We have created a standalone application which does not require internet connectivity to generate reminders for preventative maintenance. We have finished the designing and building an application for the preventative maintenance reminders of 'A' vehicles and 'B' vehicles.



Dr. Arnab Hazra

Signature(s) of the student(s)

Signature of PS Faculty

Date 15th July, 2022

Date 15th July, 2022

TABLE OF CONTENTS

Topic no.	Subject	Page no.
1	Introduction	6
2	Preventive Maintenance	7-8
3	Application Development	9-13
4	Conclusion	14
5	References	15
6	Glossary	16

INTRODUCTION

Software for Preventive Maintenance Reminder of 'A' vehicles and 'B' vehicles

We had been given the task of building an application which generates preventive maintenance reminders needed by group A and group B vehicles of the Indian Army. The application which we have made is standalone in nature, that is, it does not need to be connected to internet to be able to work. It can also not be connected to other devices via Bluetooth or any other such services, indicating that the transfer of data will not take place between two different devices. All the data is going to be stored locally within the device used.

The application will have a simple, clean user interface. It allows for the addition of vehicle information, service or maintenance information, and a time for a reminder. At the time provided, a scheduled notice will be generated and sent to the user's device.

This simple interface ensures no rigorous training is needed to use the application and store the information for maintenance.

As the device won't be connected to any other and will be kept in isolation we aim to reduce the vulnerability of sensitive data and the falling of information in the wrong hands.

PREVENTIVE MAINTENANCE

Preventive maintenance (PM) is the regular and routine maintenance of equipment and assets in order to keep them running and prevent any costly unplanned downtime from unexpected equipment failure. A successful maintenance strategy requires planning and scheduling maintenance of equipment before a problem occurs. A good preventive maintenance plan also involves keeping records of past inspections and the servicing of equipment.

Because of the complexity of maintaining a preventive maintenance schedule for a large amount of equipment, many companies use preventive maintenance software to organise their required preventive maintenance tasks.

Benefits of Preventive Maintenance

Preventive maintenance provides companies with several important benefits related to costs, errors, and health and safety. These benefits include:

- Improved reliability and life of equipment
- Fewer costly repairs and downtimes associated with unexpected equipment failure
- Fewer errors in operations as a result of equipment working incorrectly
- Reduced health and safety risks

Advantages of Preventive Maintenance

Preventive maintenance offers several key advantages for businesses, including:

- Improved Safety

Maintaining assets prevents potentially dangerous failure, mitigating against injury and any associated liability lawsuits.

- Greater Equipment Lifespan

By making sure equipment runs according to guidelines that will help improve the lifespan of the asset. Failing parts reduce the life of your equipment, resulting in expensive repair or replacement.

- Improved Productivity

Statistics show that poor maintenance can reduce a company's production capacity by 20%. By meeting maintenance requirements, you can prevent this fall in productivity as well as reduce downtime to enable greater efficiency and productivity.

- Reduced Costs

It is estimated that running a piece of equipment to failure can cost ten times as much as performing periodic maintenance. The expense comes as a result of unexpected downtimes and repairs. By understanding the maintenance requirements, you can schedule necessary repairs or part replacements at a suitable time, whether that can be achieved internally or requires an outside professional.

- Reduced Energy Consumption

PM can also have an environmental benefit, since poorly maintained electrical assets tend to use more energy than those that are functioning correctly. Of course, there is also the financial benefit of lower energy bills as a result.

PREVENTIVE MAINTENANCE REMINDER APPLICATION

What were the requirements of the PS station?

We were asked to create a standalone application that would give users service and maintenance reminders for their vehicles. In the beginning, we concentrated on scheduling a notification feature, and subsequently added an alarm feature to the application. The user should be able to enter information such as vehicle number, service or maintenance info, and the time at which he/she should be notified. This application won't rely on the Internet to work since we stored all our data locally, that is on the device itself.

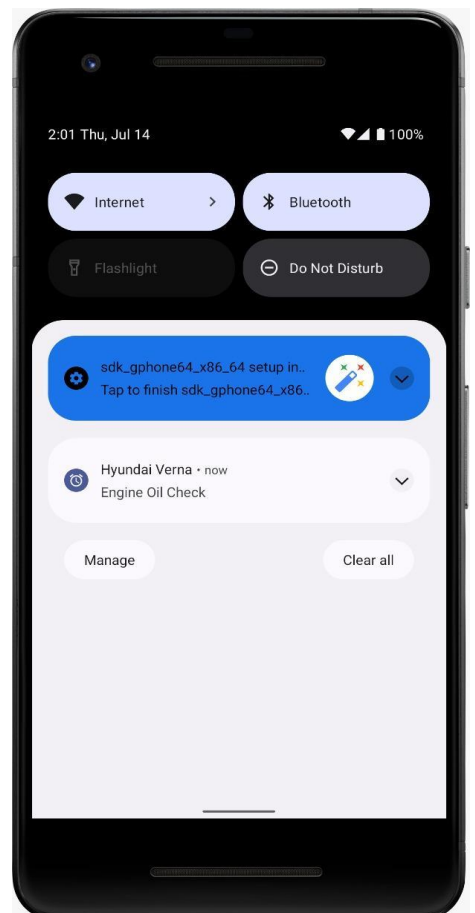
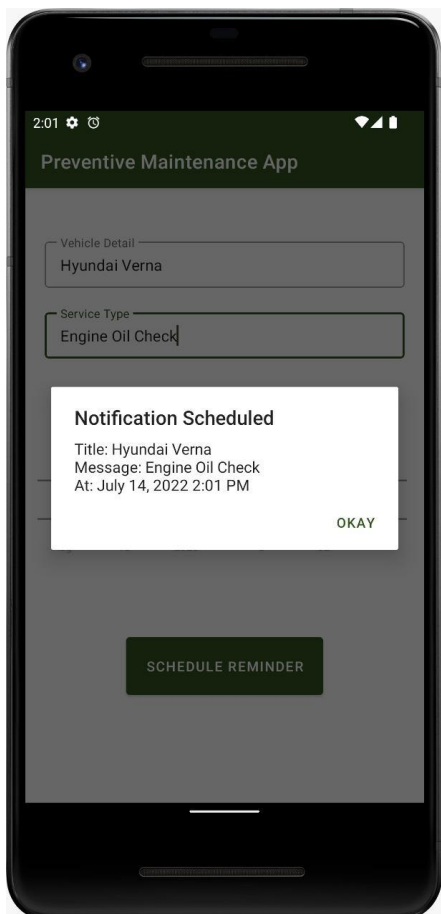
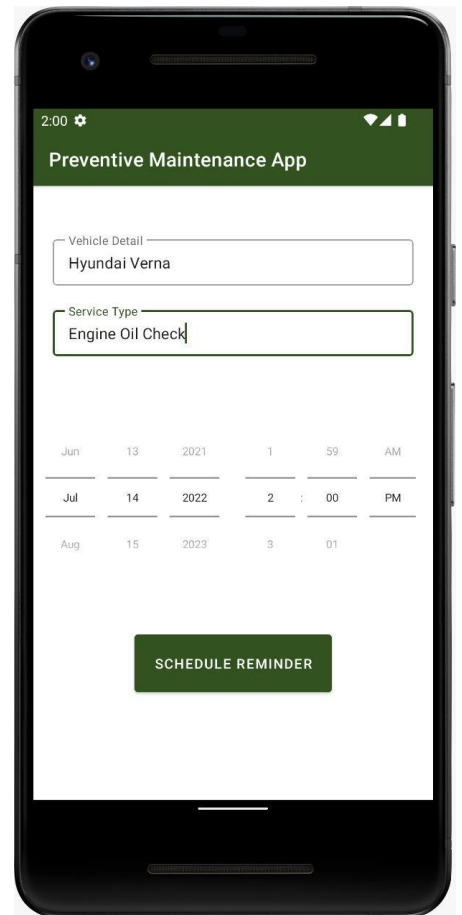
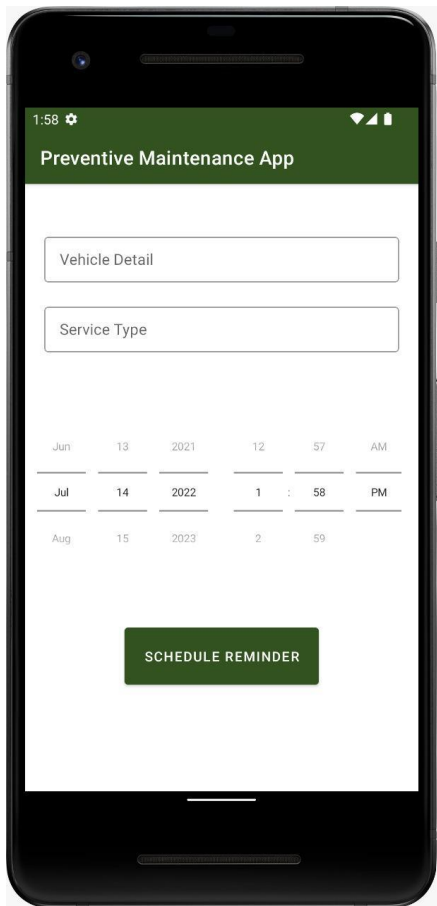
How have we developed the application?

Initially, we had to choose a language and framework that would suit our demands for this application and allow us to simply implement the various functionalities we aim to offer. React Native, Flutter, Kotlin, and Java are a few of the programming languages which can be used and there are many frameworks available that may be used to create applications. We made the decision to use Android Studio as our Integrated Development Environment and Kotlin as our development programming language. The Integrated Development Environment (IDE) we're using is called Android Studio, and it's the official IDE for Google's Android operating system. Using the IntelliJ IDEA tools from JetBrains, it was developed specifically for Android development.

We decided to use Kotlin because it is completely compatible with Java code. JetBrains created the statically typed, general-purpose programming language known as Kotlin. It is a new language for the JVM, initially presented by JetBrains in 2011. Kotlin was officially recognised as one of the recognised programming languages for Android development by Google. In addition to Kotlin, we also require the language XML. Although it is similar to HTML in structure, Extensible Markup Language (XML) lacks built-in tags. For storing, transferring, and reconstructing arbitrary data, it functions as a file format and markup language. It provides a set of instructions for encoding documents in a way that is readable by both machines and humans.

What have we built?

We have built an application that generates reminders for preventative maintenance. A reminder will be sent to the user via notice at the specified time after entering the vehicle name or ID, the service type, the reminder time, and the date.



What does the development /coding part of our application look like?

MainActivity.kt file is the main kotlin class that starts when the user clicks on the app icon.

```
private fun scheduleNotification() {
    val intent = Intent(applicationContext, Notification::class.java)
    val title : String = findViewById<TextInputLayout>(R.id.titleET).editText?.text.toString().trim()
    val message :String = findViewById<TextInputLayout>(R.id.messageET).editText?.text.toString().trim()

    //    val title= binding.titleET.toString()
    //    val message= binding.messageET.toString()
    intent.putExtra(titleExtra,title)
    intent.putExtra(messageExtra,message)

    val pendingIntent= PendingIntent.getBroadcast(
        applicationContext,
        notificationID,
        intent,
        flags: PendingIntent.FLAG_IMMUTABLE or PendingIntent.FLAG_UPDATE_CURRENT
    )

    val alarmManager=getSystemService(Context.ALARM_SERVICE) as AlarmManager
    val time = getTime()
    alarmManager.setExactAndAllowWhileIdle(
        AlarmManager.RTC_WAKEUP,
        time,
        pendingIntent
    )
    showAlert(time, title, message)
}
```

The private function `scheduledNotification()` in `MainActivity.kt` (picture pasted above) is the one that extracts text from user input and stores it in a variable. When a user clicks on the Schedule Notification button these variables are passed wrapped in an intent.

An **intent** is to perform an action on the screen. It is mostly used to start an activity, send a broadcast receiver, start services and send messages between two activities. On clicking the button, a new activity named `Notification.kt` is started that receives Notification through Broadcast receiver. A **broadcast receiver** is an Android component that allows an application to respond to messages (an Android Intent) that are broadcast by the Android operating system or by an application.

```

package com.example.localnotif

import ...

const val notificationID = 1
const val channelId = "channel1"
const val titleExtra = "titleExtra"
const val messageExtra = "messageExtra"

class Notification : BroadcastReceiver() {

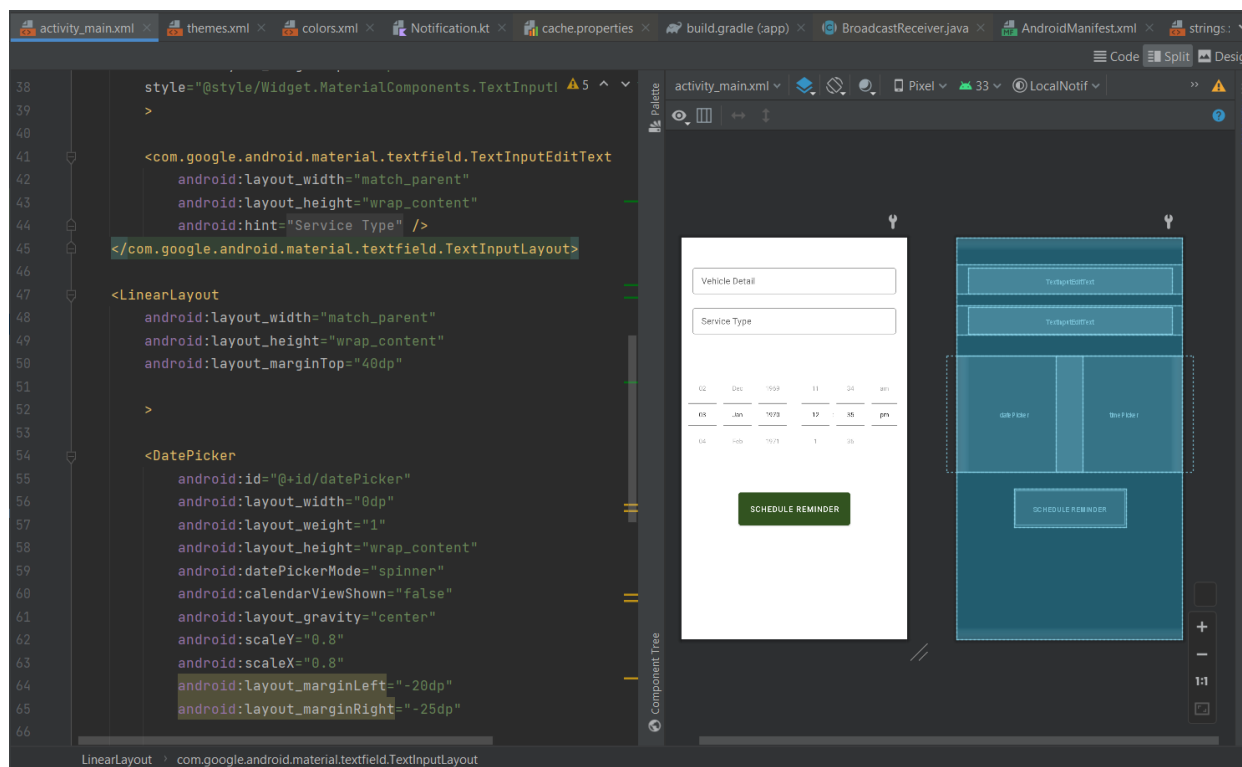
    override fun onReceive(context: Context, intent: Intent?) {

        val notification = NotificationCompat.Builder(context, channelId)
            .setSmallIcon(R.drawable.ic_baseline_access_alarm_24)
            .setContentTitle(intent?.getStringExtra(titleExtra))
            .setContentText(intent?.getStringExtra(messageExtra))
            .build()

        val manager = context.getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
        manager.notify(notificationID, notification)
    }
}

```

After the button is clicked, functioning is moved from MainActivity.kt to Notification.kt where the rest of the work of displaying notification on write time is executed.



While kotlin class files are used for functioning, the whole user interface is coded in an XML file. All the styling color, border, margins and properties are created and edited here in this file named activity_main.xml

How will our project contribute?

Our application will make a significant contribution to the Army. Automation in the maintenance field of national security will enable the reduction of manual labour in this field. Manpower will be directed to other fields and the cost of maintenance will also drastically reduce. The lifespan of each vehicle might also increase with timely intervention when required. Downtime of each vehicle will also decrease leading to a more efficient usage of these vehicles. Reactive maintenance for these vehicles will also decrease, saving essential resources for other purposes. Tampering with vehicle parts might also reduce in case it happens as the access to these vehicles will be tightly monitored.

CONCLUSION

Our application for preventive reminders not only helps users keep an organized record of services and vehicles, but it also reminds them to make sure that no vehicle misses a service that could interfere with a key Indian Army mission.

We have developed a fully functional working application with all functions integrated, and user friendliness is also our goal to make it simple to record data and receive reminders.

Every industry can be significantly impacted by software development, which also helps users by automating tedious work for them. Software development will therefore help the Indian Army by automating a number of tiresome and repetitive operations.

Regardless of how effective a system is, there is always potential for improvement. It's crucial that the system be adaptable enough to accommodate future changes. To enable the system to respond to additional changes, it has been divided into various parts. The project requirements have been covered, and every effort has been taken to make it user-friendly.

Our main aim is to make maintenance of military vehicles hassle free for the benefit of our fellow soldiers and hence for the benefit of our country.

REFERENCES

- Android development ([edureka.com](https://www.edureka.com/))
- Introduction to Kotlin([Udacity.com](https://www.udacity.com/))
- XML Introduction ([w3schools.com](https://www.w3schools.com/))
- What is Preventative maintenance? ([twi-global.com](https://www.twi-global.com/))
- List of equipment of the Indian Army([Wikipedia](https://en.wikipedia.org/))
- Extensible Markup Language(<https://www.w3.org/XML/>)
- Kotlin Docs (<https://kotlinlang.org/docs/home.html>)
- XML tags(<https://developer.mozilla.org/en-US/docs/Glossary/Tag>)

GLOSSARY

- PM – Preventative Maintenance
- XML - Extensible Markup Language
- HTML- Hyper Text Markup Language
- IDE- Integrated Development Environment
- UI- User Interface