VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590014

A VI th Semester Mini-Project Report on

"MEDICINE REMINDER APP"

Submitted in partial fulfilment of the requirements for the award of degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Prathamesh Chougule	**2AG19CS047**
Shrinath Korajkar	**2AG19CS073**
Shriram Hebbar	**2AG19CS074**
Tejashwini Rathod	**2AG19CS084**

Under the Guidance of

Prof. Gautam Dematti

Assistant Professor, Dept. of CSE,

AITM, Belagavi.

ANGADI INSTITUTE OF TECHNOLOGY & MANAGEMENT

BELAGAVI-590009

2021-2022

ANGADI INSTITUTE OF TECHNOLOGY & MANAGEMENT

BELAGAVI -590009

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Certificate

This is to certify that Project entitled **"Medicine Reminder App"** is work carried out by Prathamesh Chougule (2AG19CS047), Shrinath Korajkar (2AG19CS073), Shriram Hebbar (2AG19CS074) and Tejashwini Rathod (2AG19CSO84) in partial fulfilment of the requirements for the award of the degree of **Bachelor of Computer Science & Engineering under Visvesvaraya Technological University, Belagavi** during the year 2021-2022. It is certified that all the correction/suggestion indicated for internal assessment have been incorporated in the report. The Mobile Application Development 6th Semester Mini-Project report has been approved as it satisfies the academic requirements in respect of mini-project work prescribed for the Bachelor of Engineering degree.

Signature of the Guide	Signature of the HOD	Signature of the Principal
Prof. Gautam Dematti	Prof. Sagar Birje	Dr. Anand Deshpande
Assistant Professor,	Assistant Professor and head,	Principal and Director,
Dept. of CSE, AITM	Dept. of CSE, AITM	AITM, Belagavi

Name of the Examiner: **Signature with date:**

1._____ _____

2._____ _____

DECLARATION

We **Prathamesh Chougule (2AG19CS047), Shrinath Korajkar (2AG19CS073), Shriram Hebbar (2AG19CS074),** and **Tejashwini Rathod (2AG19CS084)** studying in the 6th semester of Bachelor of Engineering in Computer Science and Engineering at Angadi Institute of Technology and Management, Belagavi, hereby declare that this mini project work entitled **"Medicine Reminder App"** which is being submitted by us in the partial fulfillment for the award of the degree of Bachelor of Engineering in Computer Science and Engineering from Visvesvaraya Technological University, Belagavi is an authentic record of us carried out during the academic year 2021-2022 under the guidance of **Prof. Gautam Dematti,** Assistant Professor, Department of Computer Science and Engineering, Angadi Institute of Technology and Management, Belagavi.

We further undertake that the matter embodied in the dissertation has not been submitted previously for the award of any degree or diploma by us to any other university or institution.

Place: Belagavi

Date:

Prathamesh Chougule

Shrinath Korajkar

Shriram Hebbar

Tejashwini Rathod

ACKNOWLEDGEMENT

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

First and foremost, we wish to record our sincere gratitude to **Management of Angadi Institute of Technology and Management, Belagavi** and to our beloved Principal, **Dr. Anand Deshpande,** Angadi Institute of Technology and Management, Belagavi for his constant support and encouragement in preparation of this report and for making available library and laboratory facilities needed to prepare this report.

Our sincere thanks to our HOD, **Prof. Sagar Birje,** Department of Computer Science and Engineering, Angadi Institute of Technology and Management, Belagavi for his valuable suggestions and guidance throughout the period of this report.

We express our sincere gratitude to our guide **Prof. Gautam Dematti,** Assistant Professor, Department of Computer Science and Engineering, Angadi Institute of Technology and Management, Belagavi for guiding us in investigations for this seminar and in carrying out experimental work. Our numerous discussions with him were extremely helpful. We hold him in esteem for guidance, encouragement and inspiration received from him.

The mini-project on **"Medicine Reminder App"** was very helpful to us in giving the necessary background information and inspiration in choosing this topic for the project.

Last but not the least, we wish to thank our **parents** for financing our studies in this college as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

Prathamesh Chougule
Shrinath Korajkar
Shriram Hebbar
Tejashwini Rathod

ABSTRACT

The purpose of the project entitled as "Medicine Reminder App" is to develop a software which is used to set reminder for medicine intake. It is user friendly simple, fast, and cost – effective. It deals with the collection of patient's information, medical history and their medicine dosage etc. Traditionally, it was done manually. This is an Android-based application in which an automatic alarm ringing system is implemented. It focuses on doctor and patient interaction. Patients need not remember their medicine dosage timings as they can set an alarm on their dosage timings. The alarm be set for multiple medicines and timings including date, time and medicine description. System input contains medicine name, dosage, date, time and type of medicine. The system focuses on easy navigation and good user interface. Many such Medical Reminder Systems have been developed where a new hardware is required but, in our work, we have made an attempt to develop a system which is economical, time-saving and supports medication adherence. The proposed system is based on Android Operating system which will remind the users to take medicines on time through notification and automatic alarm ringing system. The scheduled reminder will not suggest any kind of medicine which is not prescribed by the doctor that will assure the safety of the patient and also will avoid wrong dosages.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

This App is the process of developing a mobile android application for all the people who take medicine on daily basis. To create this application, the development environment Android Studio, as well as the Java programming language was used. In addition to Java, with Android Studio 3.0.0 we have created an application for reminding medicine intake.

1.1 INTRODUCTION TO ANDROID

The development and application of information and communication technologies (ICT) brings major changes in all segments of the society. Expansion of the development of smart mobile devices at the beginning of the 21st century has opened a new large software market, ranging from the development of operating systems to the development of applications for various applications. The leader in the mobile operating system market is the Android platform with 74.23%, followed by the iOS platform with 20.84%, while all other mobile operating systems have a negligible small market share (1, March 2018). Various development environments are used for programming mobile applications: Firebase, iOS SDK, Visual Studio, Fabric, Android Studio and others.

It is very important to define basic knowledge and modern software packages necessary for the development of software in the field of mobile devices. As the changes in Information Technology (IT) are very fast, it is necessary that the school system at all levels is flexible and modular, so it can quickly respond to the needs of the economy and society.

Android is an OS based on Linux with a Java programming interface. It is a comprehensive open-source platform designed for mobile devices. First beta version of Android Software Development Kit (SDK) was released by Google in 2007 whereas first commercial version, Android 1.0, was released in September 2008. Android applications are usually developed in the Java language using the Android Studio Software.

- Three basic types of mobile applications are
  - Native
  - Web
  - Hybrid
- Criteria for defining types
  - Technology used for development.
  - Usage of the same application on different mobile platforms (cross-platform).

- Supported features

- Features of Android
  - Beautiful UI, Connectivity, Storage, Media support, Messaging, Web browser, multi-touch.
  - Multi-tasking, Resizable widgets, Multi-Language, GCM, Wi-Fi Direct, Android Beam.
- Development Kit
  - Once developed, Android applications can be packaged easily and sold out either through a store such as Google Play, SlideME, Opera Mobile Store, Mobango, F-droid and the Amazon Appstore.
- Tools
  - All the required tools to develop Android applications are freely available and can be downloaded from the Web. Following is the list of software's you will need before you start your Android application programming.
    1. Java JDK17 or later version
    2. Android Studio / Android SDK and Eclipse IDE for Java Developers (optional)
    3. Android Development Tools (ADT) Eclipse Plug-in (optional)

1.2 INTRODUCTION TO FRONT END SOFTWARE

1.2.1 INTRODUTION TO ANDROID STUDIO

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains IntelliJ IDEA software and designed specifically for Android development

1.2.2 INSTALLING AND RUNNING APPLICATIONS ON ANDROID STUDIO

Step 1 – System Requirements:

The required tools to develop Android applications are open source and can be downloaded from the Web. Following is the list of software's you will need before you start your Android application programming - Java JDK17 or later version Java, Android Studio.

Step 2 - Setup Android Studio:

Android Studio is the official IDE for android application development. It works based on IntelliJ IDEA, you can download the latest version of android studio from Android Studio

3.0 Download, if you are new to installing Android Studio on windows, you will find a file, which is named as android-studio-bundle-2020.3.1.22-windows.exe. So just download and run-on windows machine according to android studio wizard guideline.

If you are installing Android Studio on Mac or Linux, you can download the latest version from Android Studio Mac Download, or Android Studio Linux Download, check the instructions provided along with the downloaded file for Mac OS and Linux.

This tutorial will consider that you are going to setup your environment on Windows machine having Windows 8.1 operating system. So, let's launch Android Studio.exe, Make sure before you launch Android Studio, Our Machine should require installed Java JDK. To install Java JDK, take a references of Android environment setup installer.

You need to check the components, which are required to create applications, select Android Studio, Android SDK, Android Virtual Machine and performance (Intel chip). You also need to specify the location of local machine path for Android studio and Android SDK, we have taken default location of windows 8.1 x64 bit architecture. Then you need to specify the ram space for Android emulator by default it would take 512MB of local machine RAM.

At final stage, it would extract SDK packages into your local machine, take a while to finish the task and would take 2626MB of Hard disk space. After completing all above steps perfectly, you must click finish button and it is going to open android studio project with Welcome to android studio message.

You can start your application development by pressing start a new android studio project. In a new installation frame, it will ask Application name, package information and location of the project. After entering application name, you need to specify Minimum SDK. In our project, we have used Android 5.0. The next level will contain selecting the activity to mobile, it specifies the default layout for Applications. At the final stage it going to open development tool to write the application code.

1.2.3 INTRODUCTION TO XML

Extensible Markup Language (**XML**) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The World Wide Web Consortium's XML 1.0 Specification of 1998 and several other related specifications—all of them are free open standards.

The design goals of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human

languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

Several schema systems exist to aid in the definition of XML-based languages, while programmers have developed many Application Programming Interfaces (APIs) to aid the processing of XML data. The design goals of XML include, "It shall be easy to write programs which process XML documents".

Despite this, the XML specification contains almost no information about how programmers might go about doing such processing. The XML Info set specification provides a vocabulary to refer to the constructs within an XML document, but does not provide any guidance on how to access this information. A variety of APIs for accessing XML have been developed and used, and some have been standardized.

Existing APIs for XML processing tend to fall into these categories:

a. Stream-oriented APIs accessible from a programming language, for example SAX and StAX.
b. Tree-traversal APIs accessible from a programming language, for example DOM.
c. XML data binding, which provides an automated translation between an XML document and programming-language objects.
d. Declarative transformation languages such as XSLT and XQuery.
e. Syntax extensions to general-purpose programming languages, for example LINQ and Scala.

1.3 INTRODUCTION TO JAVA

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers "*write once, run anywhere"* (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture.

The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client-server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle) and released in 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own Hotspot Java Virtual Machine; however, the official reference implementation is the OpenJDK JVM which is free open-source software and used by most developers and is the default JVM for almost all Linux distributions.

Chapter 2

REQUIREMENT SPECIFICATIONS

2.1 SOFTWARE REQUIREMENTS

- Any 64-bit Operating System with JDK installed.
- Front End: JAVA, XML
- Tool: Android SDK with ADT Plugin
- Technology: Java Technologies
- Code-Behind Language: Java validation
- IDE: Android studio

2.2 HARDWARE REQUIREMENTS

- Quad Core Processor and Above
- Intel i5 or Ryzen 5 and above
- Speed 2.10 GHz
- RAM 8GB and Above
- 20 GB Free Hard Disk Space
- Android device

Chapter 3

OBJECTIVE OF THE PROJECT

Many Medication Systems have been developed based upon different platforms and concepts. Use of healthcare related apps is growing but there are many issues related to their functionality.

Medicine Reminder App is one of them which helps user in tracking their daily medication which is prescribed by the doctor. Here the user can set alarm for the name, type and dosage of the medicine which helps him/her easier for users to not remember every medicine prescribed to them. This reduces the improper medicine intake and reduces causality of harmful reactions.

Our app stores the user medicine information and it will provide the information about the medicine timings. It will provide the information about the medicine timings. It supports an easy implementation as it is less expensive, reliable, scalable, accessible to anyone with smartphones, and do not require separate devices, packaging or extra hardware.

Main Objectives of project

- Simple and reliable app.
- Show the medicine intakes of users.
- Simple User Interface.
- No root required.
- No Privacy Security Issues.
- Reduce the risk of improper medicine intake.

Chapter 4

IMPLEMENTATION CODE

4.1 IMPORTANT JAVA FUNCTIONS

4.1.1 MedicineActivity.java

a. public void deleteMedicineAlarm(MedicineAlarm medicineAlarm, Context context) {

```
List<MedicineAlarm> alarms =
        mMedicineRepository.getAllAlarms(medicineAlarm.getPillName());
for (MedicineAlarm alarm : alarms) {
        mMedicineRepository.deleteAlarm(alarm.getId());
        Intent intent = new Intent(context, ReminderActivity.class);
        intent.putExtra(ReminderFragment.EXTRA_ID, alarm.getAlarmId());
        PendingIntent operation = PendingIntent.getActivity(context,alarm.getAlarmId(),
                                        intent,PendingIntent.FLAG_UPDATE_CURRENT);
        AlarmManager alarmManager = (AlarmManager)
                Objects.requireNonNull(context).getSystemService(ALARM_SERVICE);
        if (alarmManager != null) {
                alarmManager.cancel(operation);
        }
}
mMedView.showMedicineDeletedSuccessfully();
}
```

b. private void processMedicineList(List<MedicineAlarm> medicineAlarmList) {

```
if (medicineAlarmList.isEmpty()) {
 mMedView.showNoMedicine();
} else {
Collections.sort(medicineAlarmList);
mMedView.showMedicineList(medicineAlarmList);
}
}
```

4.1.2 AddMedicineActivity.java

a. public void onClick(View v) {

```
int checkBoxCounter = 0;
String pill_name = editMedName.getText().toString();
String doseQuantity = tvDoseQuantity.getText().toString();
Calendar takeTime = Calendar.getInstance();
Date date = takeTime.getTime();
```

```
String dateString = new SimpleDateFormat("MMM d, yyyy",
                                Locale.getDefault()).format(date);
MedicineAlarm alarm = new MedicineAlarm();
int alarmId = new Random().nextInt(100);
if (!mPresenter.isMedicineExits(pill_name)) {
        Pills pill = new Pills();
        pill.setPillName(pill_name);
        alarm.setDateString(dateString);
        alarm.setHour(hour);
        alarm.setMinute(minute);
        alarm.setPillName(pill_name);
        alarm.setDayOfWeek(dayOfWeekList);
        alarm.setDoseUnit(doseUnit);
        alarm.setDoseQuantity(doseQuantity);
        alarm.setAlarmId(alarmId);
        pill.addAlarm(alarm);
        long pillId = mPresenter.addPills(pill);
        pill.setPillId(pillId);
        mPresenter.saveMedicine(alarm, pill);
} else { // If Pill already exists
        Pills pill = mPresenter.getPillsByName(pill_name);
        alarm.setDateString(dateString);
        alarm.setHour(hour);
        alarm.setMinute(minute);
        alarm.setPillName(pill_name);
        alarm.setDayOfWeek(dayOfWeekList);
        alarm.setDoseUnit(doseUnit);
        alarm.setDoseQuantity(doseQuantity);
        alarm.setAlarmId(alarmId);
        pill.addAlarm(alarm);
        mPresenter.saveMedicine(alarm, pill);
}
List<Long> ids = new LinkedList<>();
 try {
        List<MedicineAlarm> alarms = mPresenter.getMedicineByPillName(pill_name);
        for (MedicineAlarm tempAlarm : alarms) {
                if (tempAlarm.getHour() == hour && tempAlarm.getMinute() == minute) {
                        ids = tempAlarm.getIds();
                         break;
                }
```

```
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
            for (int i = 0; i < 7; i++) {
                if (dayOfWeekList[i] && pill_name.length() != 0) {
                    int dayOfWeek = i + 1;
                    long _id = ids.get(checkBoxCounter);
                    int id = (int) _id;
                    checkBoxCounter++;
                    Intent intent = new Intent(getActivity(), ReminderActivity.class);
                    intent.putExtra(ReminderFragment.EXTRA_ID, _id);
                    PendingIntent operation = PendingIntent.getActivity(getActivity(), id, intent,
                                                    PendingIntent.FLAG_UPDATE_CURRENT);
                    AlarmManager alarmManager = (AlarmManager)
                Objects.requireNonNull(getActivity()).getSystemService(ALARM_SERVICE);
                    Calendar calendar = Calendar.getInstance();
                    calendar.set(Calendar.HOUR_OF_DAY, hour);
                    calendar.set(Calendar.MINUTE, minute);
                    calendar.set(Calendar.SECOND, 0);
                    calendar.set(Calendar.MILLISECOND, 0);
                    calendar.set(Calendar.DAY_OF_WEEK, dayOfWeek);
                    long alarm_time = calendar.getTimeInMillis();
                    if (calendar.before(Calendar.getInstance()))
                        alarm_time += AlarmManager.INTERVAL_DAY * 7;
                    assert alarmManager != null;
                    alarmManager.setRepeating(AlarmManager.RTC_WAKEUP, alarm_time,
                                    AlarmManager.INTERVAL_DAY * 7, operation);
                }
            }
            Toast.makeText(getContext(), "Alarm for " + pill_name + " is set successfully",
                                                    Toast.LENGTH_SHORT).show();
            showMedicineList();
        }
```

4.1.3 ReminderActivity.java

a.
```
public void showMedicine(MedicineAlarm medicineAlarm) {
    this.medicineAlarm = medicineAlarm;
    mVibrator = (Vibrator) getContext().getSystemService(VIBRATOR_SERVICE);
    long[] pattern = {0, 1000, 10000};
```

```
        mVibrator.vibrate(pattern, 0);
        mMediaPlayer = MediaPlayer.create(getContext(), R.raw.cuco_sound);
        mMediaPlayer.setLooping(true);
        mMediaPlayer.start();
        tvMedTime.setText(medicineAlarm.getStringTime());
        tvMedicineName.setText(medicineAlarm.getPillName());
        tvDoseDetails.setText(medicineAlarm.getFormattedDose());
    }
```

b.
```
    private void onMedicineTaken() {
        History history = new History();
        Calendar takeTime = Calendar.getInstance();
        Date date = takeTime.getTime();
        String dateString = new SimpleDateFormat("MMM d, yyyy",
                                    Locale.getDefault()).format(date);
        int hour = takeTime.get(Calendar.HOUR_OF_DAY);
        int minute = takeTime.get(Calendar.MINUTE);
        String am_pm = (hour < 12) ? "am" : "pm";
        history.setHourTaken(hour);
        history.setMinuteTaken(minute);
        history.setDateString(dateString);
        history.setPillName(medicineAlarm.getPillName());
        history.setAction(1);
        history.setDoseQuantity(medicineAlarm.getDoseQuantity());
        history.setDoseUnit(medicineAlarm.getDoseUnit());
        presenter.addPillsToHistory(history);
        String stringMinute;
        if (minute < 10)
           stringMinute = "0" + minute;
        else
           stringMinute = "" + minute;
        int nonMilitaryHour = hour % 12;
        if (nonMilitaryHour == 0)
           nonMilitaryHour = 12;
        Toast.makeText(getContext(), medicineAlarm.getPillName() + " was taken at " +
                        nonMilitaryHour + ":" + stringMinute + " " + am_pm + ".",
                         Toast.LENGTH_SHORT).show();
        Intent returnHistory = new Intent(getContext(), MedicineActivity.class);
        startActivity(returnHistory);
        getActivity().finish();
    }
```

c. private void onMedicineIgnored() {

```
History history = new History();
Calendar takeTime = Calendar.getInstance();
Date date = takeTime.getTime();
String dateString = new SimpleDateFormat("MMM d, yyyy",
                                    Locale.getDefault()).format(date);
int hour = takeTime.get(Calendar.HOUR_OF_DAY);
int minute = takeTime.get(Calendar.MINUTE);
String am_pm = (hour < 12) ? "am" : "pm";
history.setHourTaken(hour);
history.setMinuteTaken(minute);
history.setDateString(dateString);
history.setPillName(medicineAlarm.getPillName());
history.setAction(2);
history.setDoseQuantity(medicineAlarm.getDoseQuantity());
history.setDoseUnit(medicineAlarm.getDoseUnit());
presenter.addPillsToHistory(history);
String stringMinute;
if (minute < 10)
   stringMinute = "0" + minute;
else
   stringMinute = "" + minute;
int nonMilitaryHour = hour % 12;
if (nonMilitaryHour == 0)
   nonMilitaryHour = 12;
Toast.makeText(getContext(), medicineAlarm.getPillName() + " was ignored at " +
nonMilitaryHour + ":" + stringMinute + " " + am_pm + ".", Toast.LENGTH_SHORT).show();
Intent returnHistory = new Intent(getContext(), MedicineActivity.class);
startActivity(returnHistory);
getActivity().finish();
}
```

4.1.4 MonthlyReportActivity.java

a. private void loadHistoryFromDb(final boolean showLoading) {

```
if (showLoading) {
        mMonthlyReportView.setLoadingIndicator(true);
}
mMedicineRepository.getMedicineHistory(new
         MedicineDataSource.LoadHistoryCallbacks() {
        @Override
```

```
                public void onHistoryLoaded(List<History> historyList) {
                        List<History> historyShowList = new ArrayList<>();
                        for (History history : historyList) {
                                switch (mCurrentFilteringType) {
                                case ALL_MEDICINES:
                                        historyShowList.add(history);
                                    break;
                                  case TAKEN_MEDICINES:
                                        if (history.getAction() == 1) {
                                           historyShowList.add(history);
                                        }
                                        break;
                                  case IGNORED_MEDICINES:
                                        if (history.getAction() == 2) {
                                           historyShowList.add(history);
                                        }
                                    break;
                                }
                        }
                        processHistory(historyShowList);
                         if (!mMonthlyReportView.isActive()) {
                                return;
                        }
                    if (showLoading) {
                        mMonthlyReportView.setLoadingIndicator(false);
                    }
                }
                @Override
                public void onDataNotAvailable() {
                        if (!mMonthlyReportView.isActive()) {
                                return;
                        }
                    if (showLoading) {
                        mMonthlyReportView.setLoadingIndicator(false);
                    }
                        mMonthlyReportView.showLoadingError();
                }
        });
}
```

Chapter 5

SNAPSHOTS

5.1 HOME PAGE

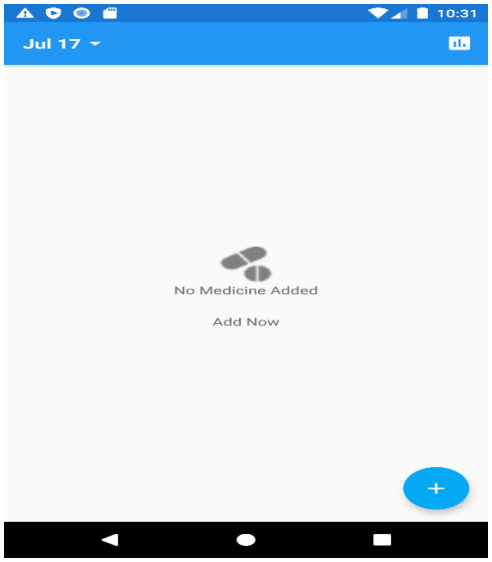

Figure 5.1.1 Home Page of the App

The above Image shows the Home Page for the Medicine Reminder App which contains list of all the active medicines and options to add new medicine and view history.

5.2 ADD NEW MEDICINE

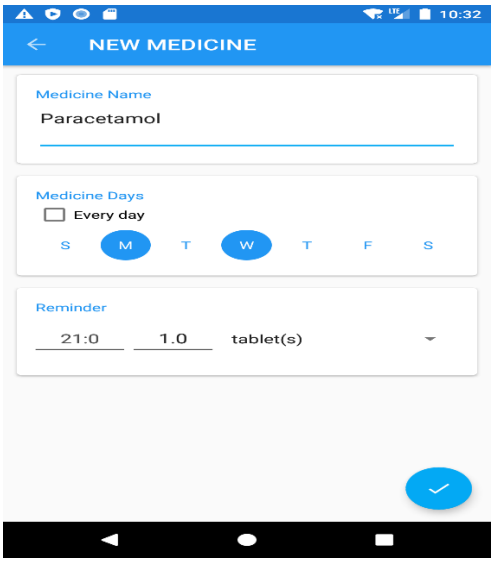

Figure 5.2.1 Page to add new Medicine

This is the page to add new medicines which contains days, time, amount and type of medicine that user want to take.

5.3 SET TIME OF MEDICINE

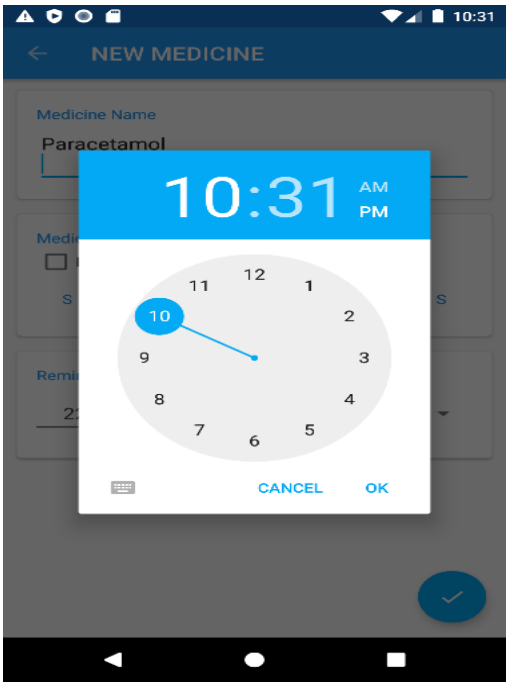

Figure 5.2.1 Page to set time of Medicine

We can set the time of medicine using analog as well as digital clock also am and pm.

5.4 ACTIVE MEDICINES

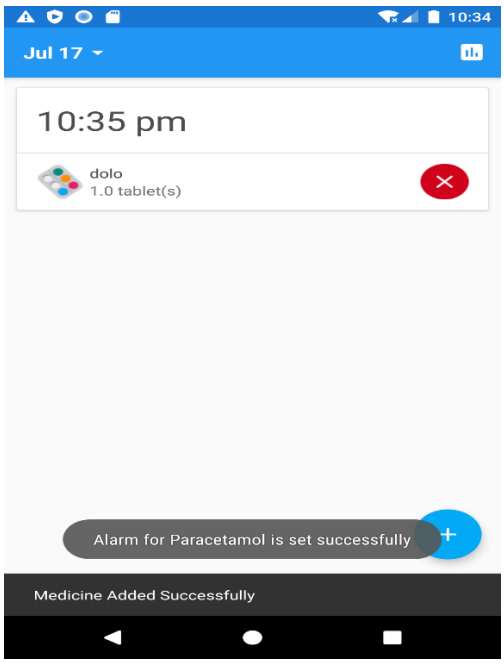

Figure 5.2.1 List of Active Medicines

After we add new medicine, we can see the list of added medicines and upcoming medicines in this page.

5.3 MEDICINE ALARM

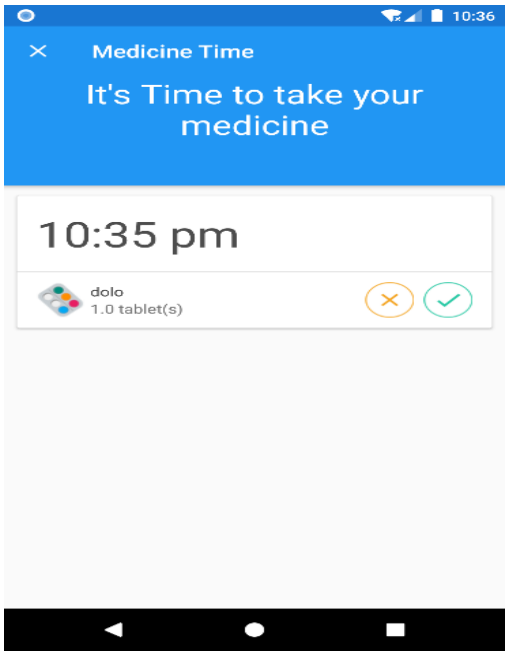

Figure 5.2.1 Alarm for the Medicine

This Page Shows the interface user will get when the alarm for the specified medicine will trigger. The phone will vibrate along with the sound in background.

5.3 MEDICINE HISTORY

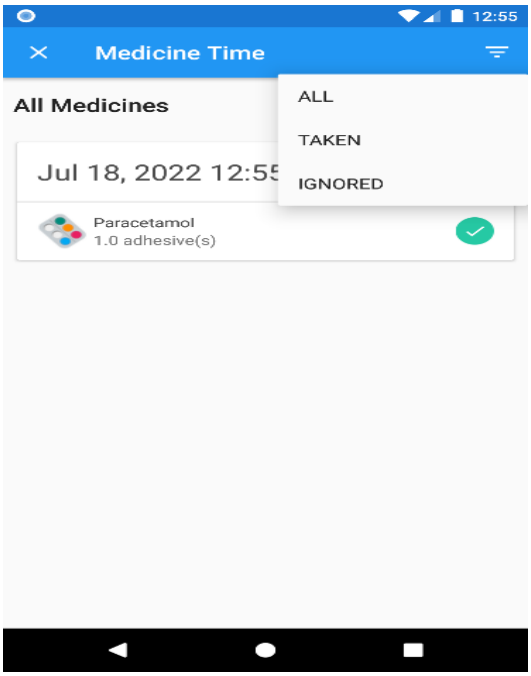

Figure 5.2.1 List of taken and ignored Medicines

The above show image is the interface where user will get the history of all the taken and ignored medicines that user has set in the past.

Chapter 6

CONCLUSION

The Medicine Reminder application gives reliable reminders, good user interface, nice user experience and it supports many new features supporting medication adherence. The users will get the schedule of medicine in-take time with medicine description, starting and ending date of medicine, notification through message or email, automatic alarm ringing system and navigation system. The youths are very much concerned with the new health care awareness and are interested in knowing about new medical techniques being developed every day. So, this feature was found useful to the youths. Hence the overall system served well in our survey and it truly supports Medication Adherence. In the given work an attempt has been made to implement a system which is economical, easily accessible and improves medication adherence. The scheduled reminder will not suggest any kind of medicine which is not prescribed by the doctor that will assure the safety of the patient and also will avoid wrong dosages. We plan to focus on improving the overall performance of the system. Also, interaction between patients and doctors through video calling and secure prescription will be focused upon. Some more ways to achieve medication adherence will be focused.

REFERENCES

1. Dawn Griffiths, “Head First Android Development”
2. Michael Burton, “Android App Development for Dummies”
3. Ed Burnette, “Hello, Android Introducing Googles Mobile Development Platform”
4. Brain Hardy, “Android Programming the Big Nerd”
5. Google Developer Training, “Android Developer Fundamentals Course – Concept Reference”