Flutter Medicine Tracker and Notification App

MedTrack – Smart Pill Reminder

Background/ Problem Statement

1) In today's busy world, everyone has many responsibilities and faces a lot of stress. Because of this, people can easily get sick or develop health problems. To stay healthy, it is very important for patients to take their medicines regularly and at the right time.
2) When a patient is at home, family members or caretakers can help remind them to take their medicines on time. But what happens if the patient is not at home? For example, if the patient is traveling, staying in another city, or far away from their family, it becomes very difficult for family members to remind them about their medicine doses every time.
3) One of the biggest problems patients face is forgetting to take the correct medicine in the right amount and at the right time. This is called **medication non-adherence**. When patients do not follow their prescribed medicine schedule properly, it can lead to serious health problems. It can also increase medical expenses because the illness might get worse or require more treatment.
4) Many studies show that forgetting or skipping medicines is a common problem and can be harmful. That's why it is important to have a simple and reliable way to remind patients to take their medicines on time.
5) To solve this problem, we have developed a **Medicine Tracker and Notification app** using the Flutter framework. This app helps users keep track of their medicine schedules easily. It sends notifications or alerts when it is time to take a medicine, so the user does not forget. This way, the patient can manage their medicines properly, no matter where they are.
6) The app is written in **Dart**, a programming language created by Google. Flutter uses Dart and allows us to build apps that work on both Android and iOS devices from a single codebase. This makes the app accessible to many users. The data about medicines and reminders is stored securely using an **MSSQL database** to keep the information safe and organized.
7) In summary, this project aims to help patients take their medicines regularly and avoid missing doses, which is very important for good health and faster recovery.

Working of the Project

The Medicine Tracker and Notification system works by allowing users to manage medication details their in a simple and organized way.First, the user needs to register an account to access the system securely. After registration, the user can log in to their personal account where all their medicine information will be saved.

Managing Medicine Details

Once logged in, the user can add details about their medicines, such as the medicine name, dosage, and quantity purchased. They can also update this information whenever needed, for example, if they buy more medicine or change their prescription.

Setting Medicine Timings

The user can set or update the specific times when they need to take each medicine. This helps create a personalized medicine timetable or schedule that tracks all doses throughout the day.

Tracking Medicine Consumption

The system keeps a record of how much medicine the user has taken and how much is left. The user can view this information at any time to monitor their consumption.

Notifications and Alerts

One of the key features of the system is its notification service. The app will send reminders or alerts to the user:
When it is time to take their medicine, ensuring they never miss a dose.When the quantity of medicine is running low, so the user knows when to buy more.

Importance of Timely Medication

Taking medicines on time is very important for effective treatment and avoiding health complications. Missing or delaying doses can cause serious problems. This system helps patients take their medicines regularly by sending timely reminders and tracking consumption accurately.

Advantages

- **Easy to Maintain:**
  The system is simple to update and manage, allowing users to easily add or change their medicine information without any hassle.
- **User-Friendly Interface:**
  The app is designed with a clean and intuitive interface, making it easy for people of all ages to use, even those who are not very familiar with technology.
- **Timely Medicine Reminders:**
  One of the most important benefits is that the system sends notifications to remind users to take their medicines on time. This helps prevent missed or delayed doses, improving the user's health
- **Tracks Regular Medicine Consumption:**
  The app helps users keep a record of their daily medicine intake. This allows users to monitor how consistently they are taking their medicines and maintain a proper schedule.
- **Monitors Medicine Purchases and Consumption:**
  Users can also track the amount of medicine they have purchased and see how much has been consumed so far. This helps them plan when to buy medicines again, avoiding running out of essential medicines unexpectedly.

System Description

1. Registration

Before using the system, the user must **register** by providing basic personal details such as their name, email, phone number, and a password.
This creates a secure user account and stores the information in the system for future access.

2. Login

After registering, the user can **log in** to the system using their **username and password**.
This ensures that only the correct person can access their medication data and reminders.

3. Profile Management

Users can view, update, or edit their **personal profile** at any time.
This may include updating contact details, medical preferences, or other personal information.

4. Change Password

For security reasons, the user has the option to **change their password** whenever they want.
This helps keep the account protected and ensures that unauthorized users cannot access the system.

5. Medicine Management

This is one of the core parts of the system.
The user can:

- **View** a list of current medications.
- **Add** new medicines to the system.
- **Update** the details of any medicine (such as dosage or frequency).
- **Delete** medicines that are no longer needed

6. Timetable Management

Users can set up a **timetable** for when each medicine should be taken. They can:

- **Create** new schedules (e.g., 8 AM, 2 PM, 9 PM).
- **Update** the timing if the prescription changes.
- **Delete** old or incorrect timing entries.

7. Inventory Management

This feature helps users keep track of their **medicine stock**.

The user can:

View how many units or tablets of each medicine they currently have.

Track how much medicine has already been consumed.

Plan when to purchase more medicine before running out.

8. Notifications & Alerts

The system sends **notifications** to the user:

- When it is time to take a medicine.
- When the medicine stock is running **low**.
- If any dose is **missed** or delayed.

These reminders help the user stay on schedule and avoid missing doses.

Project cycle

For this project, we used the **Waterfall Model**. It is one of the oldest and simplest models used in software development. In this model, each step is completed before moving to the next one—like water flowing down a staircase. There is no going back to the previous step once it is finished. This model is best when the project requirements are clear from the beginning.

Steps in the Waterfall Model:

1. Requirement Gathering

- We first collected information about what the app should do.
- Example: Remind users to take medicine, track doses, send notifications.

2. System Design

- Next, we planned how the app would work.
- We designed screens, chose Flutter for development, and MSSQL for storing data.

3. Implementation (Coding)

- We started writing code to build the app.
- Features like user registration, medicine input, and reminders were created.

4. Testing

- After building the app, we checked for errors.
- We tested notifications, medicine tracking, and login features.

5. Deployment

- Once everything worked well, we installed the app on phones for real use.

6. Maintenance

- After users start using the app, we fix bugs or make updates if needed.

Sample code for notification systeym

```
import 'package:flutter_local_notifications/flutter_local_notifications.dart';


class NotificationService {

  final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin =
FlutterLocalNotificationsPlugin();


  Future<void> init() async {

    const AndroidInitializationSettings initializationSettingsAndroid =

        AndroidInitializationSettings('app_icon');


    final InitializationSettings initializationSettings = InitializationSettings(

      android: initializationSettingsAndroid,

    );


    await flutterLocalNotificationsPlugin.initialize(initializationSettings);

  }
```

```
  Future<void> scheduleNotification(int id, String title, String body, DateTime scheduledTime)
async {
    await flutterLocalNotificationsPlugin.zonedSchedule(
      id,
      title,
      body,
      scheduledTime,
      const NotificationDetails(
        android: AndroidNotificationDetails('med_track', 'Medicine Tracker'),
      ),
      androidAllowWhileIdle: true,
      uiLocalNotificationDateInterpretation:
          UILocalNotificationDateInterpretation.absoluteTime,
    );
  }
}
```

Sample code

```
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:timezone/data/latest.dart' as tz;
import 'package:timezone/timezone.dart' as tz;
final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin =
    FlutterLocalNotificationsPlugin();
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  tz.initializeTimeZones();
  final String timeZone = DateTime.now().timeZoneName;
  tz.setLocalLocation(tz.getLocation(timeZone));
  const AndroidInitializationSettings androidInit =
      AndroidInitializationSettings('@mipmap/ic_launcher');
  final InitializationSettings initSettings =
      InitializationSettings(android: androidInit);
  await flutterLocalNotificationsPlugin.initialize(initSettings);
  runApp(MedTrackApp());
}

class MedTrackApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) => MaterialApp(
        title: 'MedTrack',
        home: ReminderScreen(),
      );
}
```

```
class ReminderScreen extends StatefulWidget {
  @override
  _ReminderScreenState createState() => _ReminderScreenState();
}

class _ReminderScreenState extends State<ReminderScreen> {
  TimeOfDay selectedTime = TimeOfDay.now();
  Future<void> selectTime(BuildContext context) async {
    final TimeOfDay? picked =
        await showTimePicker(context: context, initialTime: selectedTime);
    if (picked != null) setState(() => selectedTime = picked);
  }
  Future<void> scheduleReminder(TimeOfDay time) async {
    final now = tz.TZDateTime.now(tz.local);
    final scheduled = tz.TZDateTime(
      tz.local,
      now.year,
      now.month,
      now.day,
      time.hour,
      time.minute,
    );
    if (scheduled.isBefore(now)) return;

    final androidDetails = AndroidNotificationDetails(
      'med_channel',
      'Medicine Reminder',
```

```
    channelDescription: 'Reminder to take medicine',
    importance: Importance.max,
    priority: Priority.high,
  );
  final notificationDetails = NotificationDetails(android: androidDetails);

  await flutterLocalNotificationsPlugin.zonedSchedule(
    time.hashCode,
    'Medicine Reminder',
    'Time to take your medication',
    scheduled,
    notificationDetails,
    uiLocalNotificationDateInterpretation:
        UILocalNotificationDateInterpretation.absoluteTime,
    androidAllowWhileIdle: true,
  );
}

@override
Widget build(BuildContext context) => Scaffold(
      appBar: AppBar(title: Text('Set Medicine Reminder')),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text('Reminder at: ${selectedTime.format(context)}'),
            SizedBox(height: 20),
            ElevatedButton(onPressed: () => selectTime(context), child: Text('Choose Time')),
```

```
            SizedBox(height: 10),
            ElevatedButton(
              onPressed: () => scheduleReminder(selectedTime),
              child: Text('Schedule Reminder')),
          ],
        ),
      ),
    );
}
```

Hardware Requirements

These are the minimum hardware specifications needed to develop and test the Medicine Tracker and Notification App:

Laptop or PC (for development)

Operating System:

- Windows 7 or above
- macOS Sierra or above (required for iOS development)

Processor: Intel Core i3 or higher

RAM: 8 GB or more (for smooth performance while running Android Studio and emulators)

Storage: 100 GB or more (to store code, SDKs, emulators, and logs)

Mobile Devices (for testing)

Android Phone: Running Android 6.0 (Marshmallow) or above

iPhone: Running iOS 9.0 or later (only if you plan to test on Apple devices)

Software Requirements

The following software tools are required to build and run the application successfully:

On Laptop/PC:

Android Studio (with Flutter & Dart plugin) – Main IDE for app development

Flutter SDK – Framework used for building the app

XCode (on macOS only) – Required for building/testing iOS apps

Azure Data Studio or any SQL client – Useful for managing your MSSQL