POLITECNICO DI MILANO

MSc Automation and Control Engineering

Software Engineering

Implementation Document & User Guide

For the Application of Medicine Reminder

Prof. Matteo Giovanni Rossi

Avinash Umesh Sarma 951410
Dineshkumar Ramasamy 964069
Satyabrata Dash 961528

ACADEMIC YEAR 2021/2022

Contents

1. Introduction

From studying the requirements for the medicine reminder, the Feasibility Study document provides evident justifications for developing the same. The development of the mobile applications follows different phases of documentation, starting from the Feasibility Study Document which explains the feasibility of the concept. Further, the next phase involves the study of requirements to develop the concept that makes the Requirement Analysis and Specifications Document (RASD) while the study of software architecture to follow is enclosed in the Design Document (DD). The other phases include the Implementation Document, Testing Document, and User Manual, where the information regarding the documents would be mentioned correspondingly in the later phases of documentation.

Here, this illustrates the Implementation Document (ID) and User Manual of the project of the medicine reminder abbreviated and named “MedRem”. The first half of the document particularly focuses on the implementational aspect of the application with the framework adopted. Thus, the document starts with a section on the MIT App Inventor framework and its architecture, which also gives an overview of block-based programming. In the later sections, an outlook of functions performed in each screen is given. Furthermore, this document encloses the User Manual for the users to follow, to use the mobile application.

1.1 Scope

MedRem is an easy-to-use application that helps people to keep track of their medicines by providing reminders. It has a simplistic design and vintage approach to the features. This application consists of adding, showing, modifying, and deleting the medicine information. Other than that, two kinds of reminders can be added to the medicines, namely a medicine reminder and a stock reminder. The user can set and modify them according to their intake of the medicines. Restock is yet another feature that is added to the application, allowing the user to update the current stock value.

2. MIT App Inventor

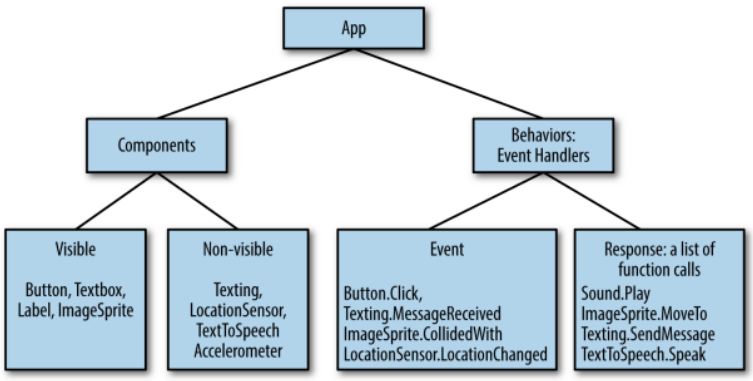
MIT App Inventor is an online development platform that anyone can leverage to solve real-world problems. It provides a web-based “What you see is what you get” (WYSIWYG) editor for building mobile phone applications targeting the Android and iOS operating systems. It uses a block-based programming language built on Google Blockly and inspired by languages such as StarLogo TNG and Scratch, empowering anyone to build a mobile phone app to meet a need.

2.1 Overview

The MIT App Inventor user interface includes two main editors: the **design editor** and the **blocks editor**. The design editor or designer is a drag-and-drop interface to lay out the application's user interface (UI) elements. The blocks editor is an environment in which app inventors can visually lay out the logic of their apps using color-coded blocks that snap together like puzzle pieces to describe the program. To aid in development and testing, App Inventor provides a mobile app called the App Inventor Companion (or just "the Companion") that developers can use to test and adjust the behaviour of their apps in real-time.

2.2 Internal Architecture

One way to describe an app's internals is to break it into two parts, its components, and its behaviors. Roughly, these correspond to the two main windows you use in App Inventor: you use the Component Designer to specify the objects (components) of the app, and you use the Blocks Editor to program how the app responds to the user and external events (the app's behavior).



Internal architecture of the App Inventor's app

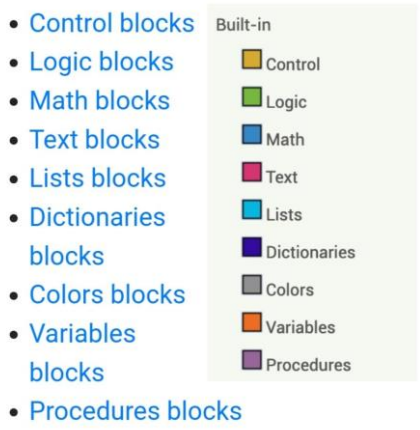
2.3 Components

There are two main types of components in an app: visible and non-visible. The app's visible components are those that you can see when the app is launched—**buttons**, **text boxes**, and **labels**. These are often referred to as the app's user interface. Non-visible components are those that you can't see, so they're not part of the user interface. Instead, they provide access to the built-in functionality of the device. Both visible and non-visible components are defined by a set of properties.

2.4 Behavior

App components are generally straightforward and easy to understand: a text box is for entering information, a button is for clicking, and so on. The behavior defines how the app should respond to events, both user-initiated (e.g., a button clicks) and external (e.g., an SMS text arriving on the phone). Fortunately, App Inventor provides a high-level blocks-based language for specifying behaviors. The blocks make programming behaviors more like plugging puzzle pieces together, as opposed to traditional text-based programming languages, which involve learning and typing vast amounts of code.

Blocks are coloured to easily identify the functionalities mentioned as a legend below.



2.5 Extensions

Extensions are external libraries that can be used in the MIT App Inventor to call procedures or functions inherent to the extension. In this application, some of the extensions are used to implement certain functionalities, that includes the following:

1. Background Tasks Extension 3.8A
   An extension to create components, call functions or tasks in the background, and when the app is alive with various features
2. ItoO – Background Tasks
   ItoO Tasks is a specially built extension that makes such an Environment that it would be possible to allow things like Procedures/Variables to be accessed when the App is In the Background.
3. Alarm Manager Extension with Notification or AutoStart
   An extension to manage the multiple repeating alarms in the background along with notifications.
4. Notification Style
   An extension to create notifications in different methods and styles.

3. Implementation

3.1. Overview

The Design document briefs about the domain architecture through the UML Diagram describing the classes. These classes are further realized in the form of different screens that makes the entire application. The application has been implemented in 5 screens namely, Home_Screen, Add_Medicine, Set_Reminders, Show_Medicine, and Modify_Medicine screen. The screens communicate with each other using the Tiny DB as a medium. With the global visibility of the database, the data could be viewed all over the application's interface. By calling the respective tag name of Tiny DB, we could get the necessary details everywhere on the application screen. Apart from the TinyDB, there are also other invisible components used in various screens that helps in the implementation that includes notifiers, clocks and some of the extensions mentioned in the section of MIT App Inventor.

Screen No.	Screens
1	Home_Screen
2	Add_Medicine
3	Set_Reminders
4	Show_Medicine
5	Modify_Medicine

3.2. Requirements and Implementation

Regarding the requirements mentioned in the Requirement Analysis and Specification Document (RASD), the application is built according to it, satisfying most of them. In reference to the table for Requirements, it shows the screens in which the corresponding requirements have been implemented. Further, descriptions of the implementation on each screen have been mentioned in different sections later in the document. The descriptions give a better outlook about the function and operations that have been used that are again connected with screens and tabulated. This can be again used to identify the functions associated with each requirement.

From the implementational aspect, all requirements have been worked except with the completion of alarms and notifications. For the moment, only notifications are called when the user wants to get reminded. While it is not exactly a requirement, there should also be an alarm that rings along with the notification. As of now, notification is implemented through the extension of Taifun Alarm Manager with no alarm sounds though, which proves to give the most stable results without latency or any other issues. The implementation of the same was also tried using various other extensions,

one of such as the Background Task Extension that integrates any number of components for one background task in repeat. Since the alarms or notifications do not adhere to the set time, this option was also dropped out. Thus, implementations of alarms have been withdrawn and can be taken as part of later releases of the app or even as future work.

No.	Requirements	Implemented Screens
1	System allows users to add medicines and their related	2
2	System displays detailed information on medicines to users	4
3	System allows users to modify the information about	5
4	System allows users to delete medicines.	4
5	System allows the user to turn on/off the medicine reminder	3,5
6	System allows users to set reminders for their medicines	3
7	System allows users to change the time of the reminders for	5
8	System allows the user to turn on/off the stock reminder	3,5
9	System allows users to account for the stock of medicines	3,5
10	System allows users to set reminders for stock of their	3,5
11	System allows users to modify the stock reminder	5
12	System allows users to modify the quantity of stock	5

3.3. Home_Screen

The Home screen is equipped with two functions that can be accessed by clicking their respective buttons.

Add Medicine:

When the user clicks on the button, they will be routed to the Add_Medicine screen. Here, the user will be able to add the medicine details and the data is stored in a database with assigned tag names.

Show Medicine:

When the user clicks on the button, the input is checked on the database if it is empty or filled. If the database returns empty, the user is notified to add the medicine. Else it will direct the user to a screen associated with the listpicker where they would be able to pick the data by clicking on it. Before picking the medicine, the list picker elements are set to the medicine names called from the database. After picking the medicine, the index value of the medicine is saved in the database

under the tag name of 'Index' and takes the user to the Show_Medicine screen. The Index value in the database, is further used in other operations like the show, modify or delete medicine.

3.4. Add _Medicine

The Add medicine screen consists of fields like name, dosage, and frequency of the medicine. All the fields on the screen are mandatory with an *Empty_Fields* check and if left blank, the user is notified to fill them and wouldn't be allowed to proceed.

The inputs from the user are stored in a temporary database which helps proceed further by clicking on the 'Set Reminder' button. After that, the medicine name in the temporary database is compared with the medicine names present in the database. If the output comes to be true, the user is notified that the medicine is already existing and need to modify some fields. Otherwise, the medicine will be saved in the database under a temporary tag that is retrieved in the Set_Reminder Screen. On clicking the button of Set Reminder, the user would be taken to the screen of Set_Reminder while Back Press or Cancel returns the user back to the Home_Screen.

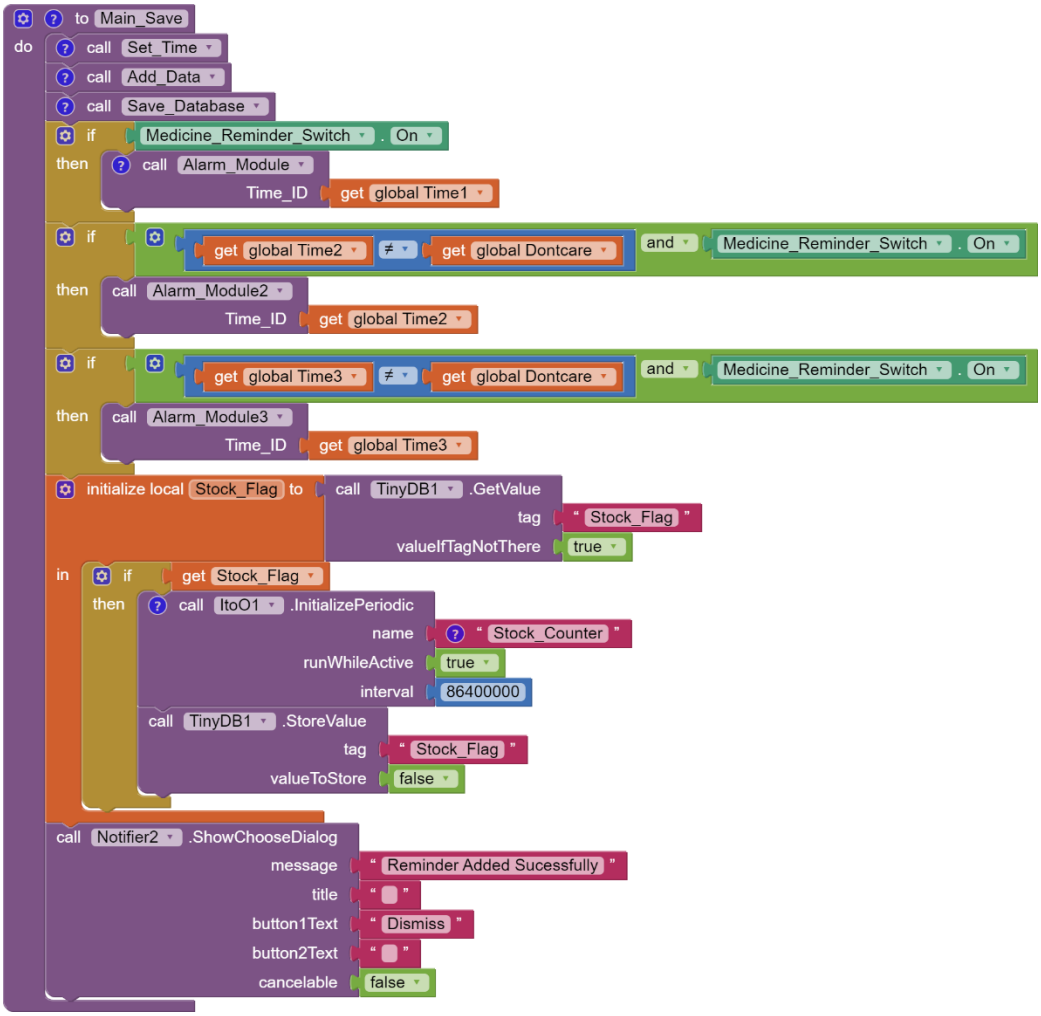
3.5. Set_Reminders

The global variables of Name, Dosage, Frequency, Time picker1, Time picker2, Time picker3, Stock, and Minimum value is initialized to create empty lists as part of the screen initialization. On calling the function of *Call_Database*, all the global variables are set to the values from the database from its corresponding tag names. Also, the screens are initialized according to the layout adjustments, which are mentioned in the later section of the document.

When the user clicks on the save button, *Empty_Fields_Set_time & Empty_Fields_Stock* checks are implemented for medicine reminders and stock reminders respectively. The checks return a false condition if the user-input fields are not empty, to proceed with further operations, mainly the *Main_Save* Function. Else, it calls a notifier to alert the user for empty fields in either medicine reminders or stock reminders with the state of respective reminders.

Referring to the snippet of code on *Main_Save*, it is composed of three functions such as *Set_Time, Add_data & Save_Database*. The function *Set_Time* gets the user-set time from the corresponding timepickers which are pre-set with certain values to enhance the user experience and convert it to a four-digit number. Further, the *Set_Time* function contains another operation known as *Time_Check*, where it checks if the values entered by the user conflict with other reminders for different medicines, ensuring its uniqueness to reuse them as ID for Alarms and Notifications. If there are coincidences of values, then the current time for the reminder is increased by 1 minute or else it is set as the user-defined value on global variables correspondingly for each timepickers. Setting the values also depend on the state of medicine reminder as well as the frequency set by

the user, in which the variables are set to a 'Don't Care' value (9999), owing to the conditions. Similar is the case with the stock reminder, which would be set as a 'Don't Care' Value if it is off, else the values are taken from the corresponding components.



In the *Add_data* function, the user-defined values of medicine details along with reminder settings are added with the already initialized global lists of Name, Dosage, etc., which are then stored in the database using the function of *Save_Database* under similar tags. The user's choice of reminders activates either *Alarm_module* or *Stock_Counter*. The *Alarm_module* contains the blocks from the extension of Taifun Alarm Manager that helps in setting repeating alarm notifications with an interval of 24hrs, taking the 4-digit time as an input, which serves the purpose of both ID and the time. The *Stock_Counter* performs a decrementing counter function based on stock and minimum values, which also calls for the dosage and frequency of the medicine from the database, executing the operation and checking if the stock falls less than the minimum value to give a notification for low stock. In the condition of the stock falling less than or equal to 0, the stock reminder would be turned off reminding the user with a notification. This function is also called in the background even when the application is closed, making it possible with the help of

ItoO extension blocks for its repeating action with an interval of 24 hrs. The counter is turned on the first-time user adds a medicine to the application.

The successful execution of all these functions is marked by a notification that the user can acknowledge, routing them back to the Home_Screen. In the event of clicking on the cancel button, the user will be taken back to Home_Screen again. Pressing the back in the phone directs the user to Add_Medicine Screen.

3.6. Show_Medicine

As mentioned in the description on the home screen, the view medicine is interlinked with the list picker by the Selection Index based on the selection of medicine by the user and saving it in the database. The screen initializes according to layout adjustments and *Call_Database* to set all the global lists with all values related to the medicines, retrieved from the database along with the selection index. From the global lists, the details of the selected medicine are retrieved by getting the values at the selection index. Based on the values at the index, the components like textboxes or labels are set with corresponding values of Names. Dosage etc. to display it back to the user, including the reminder details with different timings or stock values, mentioning the state of reminders.

Another possibility for users to perform is clicking on the Delete button. It is followed by *Cancel_Alarm* function that cancels the alarm and notification using the four-digit time IDs unique to the medicine, with the help of Taifun Alarm Manager Extension blocks. The event will also remove the entry of the selected medicine from the global lists of medicine details and the updated list is stored back in the database by calling the function of *Save_Database.* A notifier gives an alert for the successful operation of deletion once it completes execution, directing them to Home_Screen. On clicking the Modify button, the user will be taken to the Modify_Medicine Screen while clicking the cancel button takes them to Home_Screen same as the event of Back Press.

3.7. Modify_Medicine

The screen initializes according to layout adjustments and *Call_Database* to set all the global lists with all values related to the medicines, retrieved from the database along with the selection index. Also, the text boxes, timepickers, or any other components of inputs will be preloaded with the values of medicine details that are already stored in the database at the index selected by the user.

According to the requirements, the screen serves as the platform to edit any kind of medicine details whether its Name, Dosage, Frequency, or other details in regard to medicine or stock reminders in terms of either turning them On-Off or modifying their attributes. Thus, the screen

is implemented with all the back-end functions of Add_Medicine and Set_Reminder Screens indirectly.

Once the user clicks on the save button after the modification of details, the system checks for *Empty_Fields* for the medicine details, time set, or stock values depending on the states of reminders. If there are no inputs, the user would be shown with the alert to enter the input, or else the *Main_Save* Function gets activated. Similar to the *Main_Save* function in Set_Reminder Screen, the main sub-functions are *Cancel_Alarm, Set_Time, Replacing_Data, Save_Database,* and *Alarm_module*. The *Cancel_Alarm* performs the exact same operation that is mentioned in the deletion of medicine data in Show_Medicine Screen, while the *Replacing_data* function replaces the data at the particular index the user has selected with the user-modified values in the global lists and saves them in the database. The remaining functions serve the same operation as in the Set_Reminder Screen.

Apart from this, the user is also given a way to update the stock value with a restock operation, which adds the user-input values to the current stock value and replaces the same in the database. With the successful execution of all operations, the user will get a notification that needs to be acknowledged, taking them back to Home_Screen same as the cancel event. In another way, the back pressing event will result in the previous screen of Show_Medicine.

No.	Functions	Screens	Operation
1	Call_Database	3,4,5	Getting all the values from the database and setting the values to temporary lists of global variables.
2	Save_Database	3,4,5	Saving all the global variables of temporary list details in the Database.
3	Empty_Fields	2,5	Check if entered details of Name, Dosage or Frequency are empty and give an alert if conditions are not met.
4	Empty_Fields_Set_time	3,5	Check for Empty Fields in the time set and gives an alert if conditions are not met.
5	Empty_Fields_Stock	3,5	Check for Empty Fields in the Stock Values and gives alerts if conditions are not met.
6	Set_Time	3,5	Setting the Time from Timepickers and performing suitable operations to make both ID and Time.
7	Time_Check	3,5	Check if the time set is already there in the database and if there, it will add a minute to it. This is performed to reduce conflicts between simultaneous alarms.

8	Add_Data	3	Adding the Data to the temporary global list of all the details.
9	Replacing_Data	5	Replacing the Data in the temporary global list of all the medicine details.
10	Alarm_Module	3,5	Alarm module for notifications and sound, set with the time (same as ID).
11	Cancel_Alarm	4,5	Cancelling the alarms with the ID of the corresponding medicine.
12	Stock_Counter	3	Retrieves stock values from the database and decrements the value depending on the dosage and frequency, regularly.
13	Format_Time	3,4,5	Data Type Conversion from Number to String and adjusting the format of Time.

3.8. Layout Adjustments

The layouts are components that help with the arrangement of other components to fit inside them. It defines the structure of the user interface of the application. There are different kinds of layouts like **vertical**, **horizontal**, or even **scroll arrangements**, that accommodate components in their manner and hierarchy.

Concerning the screen of the application, the layouts can be made **static**, as if it doesn't change with time or based on any event triggers while it can also be made **dynamic** depending upon a particular time or about any other events or components that are controlled from the back end. In all the screens, the layouts are initialized when the screen itself is initialized and the visibility is further changed according to the required specifications.

For the application of MedRem, the configuration of layouts is kept different for each screen. The **Home Screen**, which is also screen 1 contains only a few visible components like buttons and an image, which are enclosed in a vertical arrangement that fits the entire screen. The visibility of the components doesn't change with time or any other triggers making it static, throughout the working of the application.

Similarly, screen 2 (**Add_Medicine**) which also contains labels and textboxes other than buttons in its corresponding layouts in the hierarchy, also does not depend on any other operations for its visibility.

In the **Set_Reminder** screen, the layouts enclose components such as Time Pickers and text boxes in different layouts to accommodate the number of times the user takes the medicine in a day which is obtained through frequency, input by the user in the previous screen of Add_Medicine.

The visibility of these layouts is turned on with a primary condition based on the switch for the medicine reminder, while the second condition is on the frequency that the user has selected. Similarly, the layouts for stock reminders are also controlled based on the state of the stock reminder switch.

The Show_Medicine Screen (screen 4) has a very basic configuration of layouts, that fits labels and textboxes, giving the data the user has saved in the database. Here, the visibility of layouts is adjusted according to the data that has been saved concerning the conditions of medicine reminder and stock reminder states.

The components in screen 5 (Modify_Medicine) are set in a scroll arrangement, where the user will see all the values already stored in the database before modification. The screen is a combination of Add_Medicine and Set_Reminder screens with the arrangements that have already been mentioned.

4. User Guide

4.1 Introduction

MedRem or Medicine Reminder (in short) is a mobile application that allows keeping track of your medicines systematically.

The app adheres to Android human interface guidelines for a medicine reminder app, so you should feel comfortable using it right away.

All the instructions in this user guide describe how, you as a user, use this application on your mobile phone.

The app designed is to be easy to use, so it contains the functionalities like,

- Add medicines details
- Adding medicine reminders
- Adding stock reminders
- View medicine details
- Modifying medicine details
- Deleting medicine details
- Restock medicines

4.1.1 Compatibility

MedRem application is compatible with Android phones (above Android version 4) only.

4.1.2 Help

Any questions, concerns, or issues regarding the app may be resolved by contacting the MedRem support team. Users can contact our team members by email at:

- Avinash Umesh Sarma – avinash.umesh@mail.polimi.it
- Dineshkumar Ramasamy- dineshkumar.ramasamy@mail.polimi.it
- Satyabrata Dash - satyabrata.dash@mail.polimi.it

4.2 Installation of MedRem

- On your android phone, open the link https://drive.google.com/file/d/1WYk5kzgFh826IZUyokplZjUvzLuKhTkv/view?usp=sharing
  or scan the QR Code
- Select the MedRem app in the folder
- Download the .apk file
- Follow the standard installation procedure

4.3 Application Overview

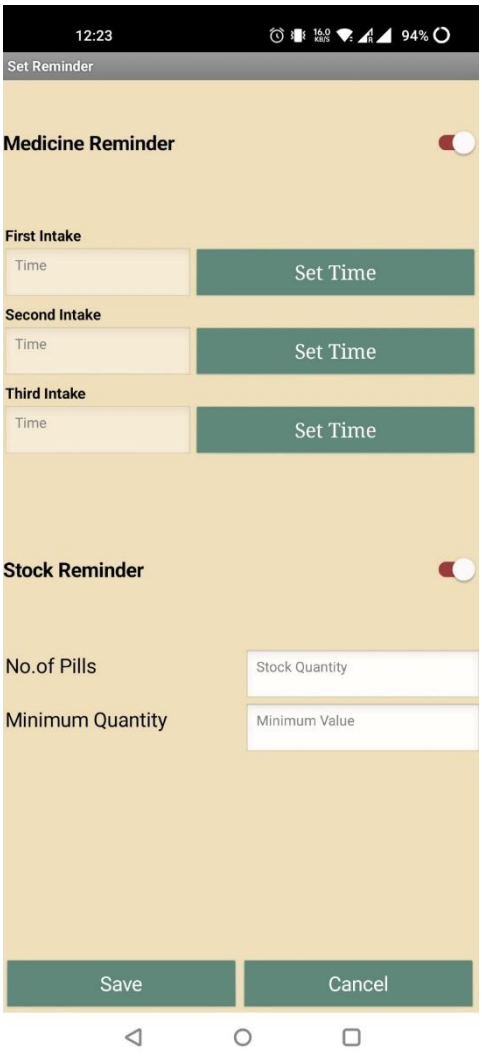
The application interface contains 5 tabs: Home, Add Medicine, Set Reminder, Show Medicine, and Modify medicine.

Home

In the 'Home' screen, you can either choose to add medicine or you can choose to show medicine if a medicine is added already.

Add Medicine

This screen contains the details of the medicine like name, frequency, and dosage. Once you have filled it, you can click on the set reminder to add reminders.

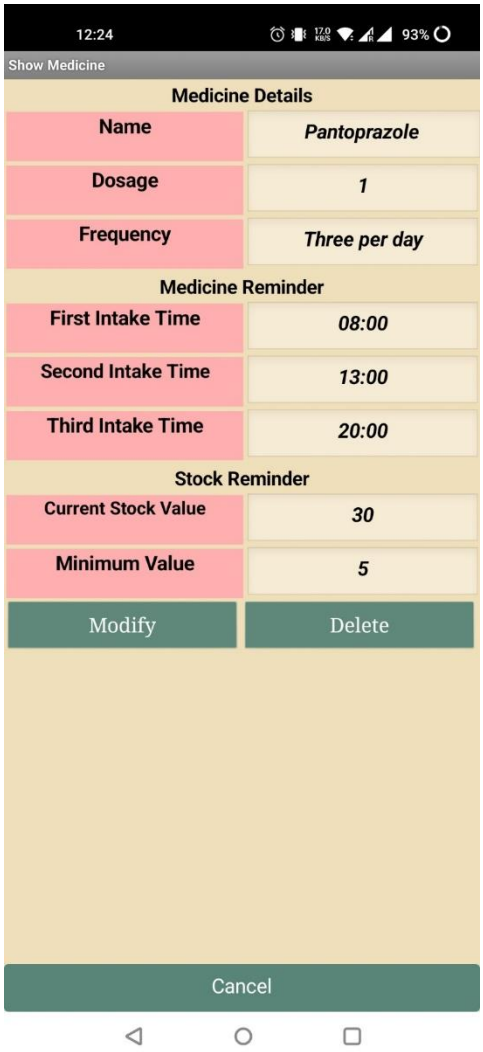


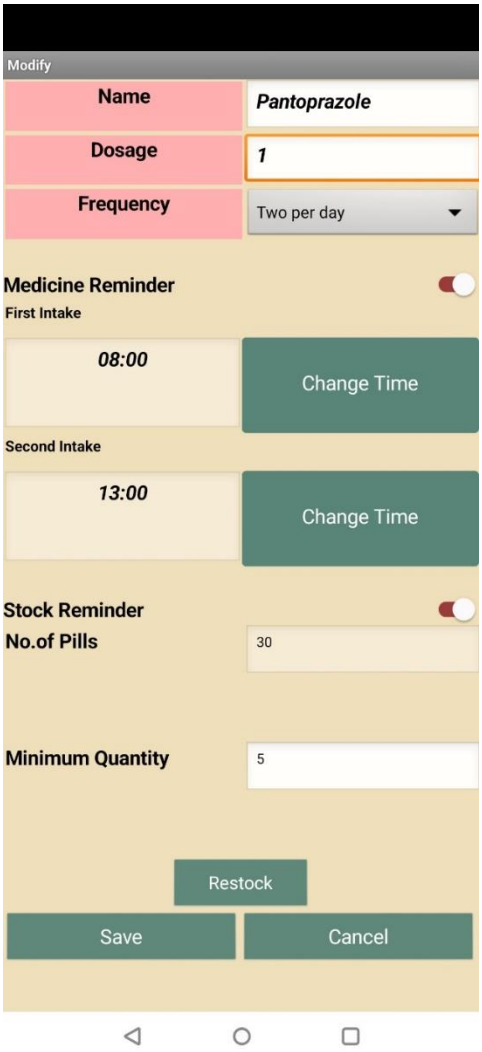
Set Reminder

The screen is linked with the set reminder button. You must toggle the medicine reminder button ON to display the timers. Based on the frequency, the timers are displayed. You can set the time but by default, it displays 08:00. Also, it contains the stock reminder button; after toggling it ON, you will be able to enter the No. of pills and Minimum quantity values. Once you fill it in, you can click the save.

Show Medicine

This screen displays all the detailed information about the medicine. From here, you can perform two major operations: Modify or Delete.





Modify medicine

On this screen, all the fields from add medicine and set reminder screens are open to modifying. You can modify any of the previously set values and save them. Not to forget, the restock button is also present. It is a part of the stock reminder and is only possible if the button is toggled ON. Once you click on the restock, it asks for a user input value. Entering the same and confirming it adds that to the current stock value.

4.4 Instructions

4.4.1 Add Medicine details

When you are opening the app for the first time as a user, please follow the given steps:

- Open Application
- Click 'Add Medicine'. You will be taken to the next screen to add medicine details.
- Enter the name, frequency, and dosage of the medicine
    - Note: Entering the details is a mandatory process. Without that, you can't move on to the next step and you will receive a notification to fill it.
- Click 'Set Reminder'.
    - Note: This will take you to the next screen for setting the reminders. If you don't want to set reminders, toggle the switches off for the reminders and proceed to the next step.
- Click 'Save'

You have saved the medicine details successfully without any reminders.
Important: If you need to add reminders, you must toggle their respective switches and their instructions are given in the further sections.

4.4.2 Set Medicine Reminders

When you have entered the medicine details, please follow the given steps:

- Click on the 'Set Reminder'. It will take you to the next screen for setting the medicine reminder.
- Toggle the 'Medicine Reminder' to ON.
- Set the time accordingly
    - Note: Setting the time for the reminder is a mandatory process if the switch is turned ON. Without that, you can't move on to the next step and you will receive a notification to fill it.
- Click 'Save'

You have set the medicine reminder for your medicine successfully.

Important: The maximum frequency allowed by the application is three. So, a maximum of 3 timers can be set for one medicine.

4.4.3 Set Stock Reminders

When you have entered the medicine details, please follow the given steps:

- Click on the 'Set Reminder'. You will be directed to the next screen for setting the stock reminder.
- Toggle the 'Stock Reminder' button to ON

- Enter the No. of pills and minimum number of pills
    - Note: Entering the details is a mandatory process. Without that, you can't move on to the next step and you will receive a notification to fill it.
- Click 'Save'

You have set the stock reminder to your medicine successfully.

4.4.4 View Medicine details

When you are on the home screen of the app, please follow the given steps:

- Click on the 'Show Medicine'. You will be directed to another screen, where the names of the medicines are in form of a list.
    - Note: When you are clicking it for the first time, it returns a notification to add a medicine.
- Click on the medicine to which you want to view the details.
- Detailed information about the medicine is displayed along with the name, dosage, and frequency.
- Depending upon the conditions set for the reminders, you can see the corresponding fields of the medicine reminder and stock reminder respectively.

You have viewed the medicine details successfully.

4.4.5 Modify Medicine

When you want to modify any details of medicine and you are on the home screen of the application, please follow the given steps:

- Click on the 'Show Medicine'. You will be directed to another screen, where the names of the medicines are in form of a list
- Click on the medicine to which you want to modify the details.
- Detailed information about the medicine is displayed.
- Click 'Modify'
- Fields like name, frequency, and dosage will be prefilled with the old data and can be modified on the same screen.
- Fields like medicine and stock reminder will also be prefilled depending upon the previous conditions set and can be modified on the same screen. If not set previously, you can set it on the same screen.
    - Note: To set the reminders, please refer to the previous section.
- Click 'Save'

You have modified the details of the medicine successfully.

4.4.6 Delete Medicine

When you want to delete the details of the medicine and you are on the home screen of the application, please follow the given steps:

- Click on the 'Show Medicine'. You will be directed to another screen, where the names of the medicines are in form of a list
- Click on the medicine to which you want to delete the details.
- Detailed information about the medicine is displayed.
- Click 'Delete'. You will receive a notification of the successful deletion.

You have deleted the details of the medicine successfully.

4.4.7 Restock Medicine

When you want to restock the medicine and you are on the home screen of the application, please follow the given steps:

- Click on the 'Show Medicine'. You will be directed to another screen, where the names of the medicines are in form of a list.
- Click on the medicine to which you want to modify the details.
- Detailed information about the medicine is displayed.
- Click 'Modify' and you will be taken to the next screen displaying all the modifiable details.
- Click on the 'Restock' under the stock reminder section.
- Enter the number of pills to be added to the stock.
- Click on the 'OK' on the mini screen.
- Click 'Save'

You have restocked the medicine successfully.

5. References

- https://community.appinventor.mit.edu/t/background-tasks-extension-3-8-a/29160
- https://community.appinventor.mit.edu/t/itoo-background-tasks-special/55125
- http://puravidaapps.com/alarmmanager.php
- https://community.appinventor.mit.edu/t/free-notification-style-extension-with-various-types-of-notification/12115
- https://assets.website-files.com/5cff53d89e57b8ab97ee595e/5d9f3993f0254882f0e4a5aa_Kind-user-guide-for-healthcare-mobile-app.pdf
- https://dl.acronis.com/u/pdf/ACM_userguide_en-US.pdf