




## Remindly: Android-Based Task Reminder App with Agile Methods to Improve Student Study Time Efficiency

Artika Sari Dewi<sup>1</sup>, Zahwa Aurellia Adha Maulana Syach<sup>2\*</sup>, Anggun Nurpadila<sup>3</sup>

<sup>1,2,3</sup> Makassar State University, Indonesia

Corresponding e-mail : [zahwaur2105@gmail.com](mailto:zahwaur2105@gmail.com)

ARTICLE INFO	ABSTRACT
<b>Keywords:</b> Agile Android Task Reminders Time Management Remindly	The rapid development of digital technology demands practical solutions in time management, especially for students and students who often face difficulties in managing academic assignments. This research aims to develop the Remindly application, an Android-based task reminder application designed using the Agile method to help users systematically record, manage, and receive task notifications. The development process is carried out in three sprints, including the stages of planning, needs analysis, interface design, implementation, and testing. The app's key features include login, registration, task addition, automatic reminders, grouping tasks by status, and user profile management. The app was tested using both black-box and white-box methods, which showed that all functionality was running as planned. The Remindly application is expected to be able to improve the time efficiency, discipline, and productivity of users in daily activities, especially in the academic context.
<b>Article History</b> Received: September 10, 2024 Revised: October 20, 2024 Accepted: November 25, 2024	This is an open access article under the <a href="#">CC BY-SA</a> license 

**To cite this article :** Author. (20xx). Title. Information Technology Education Journal, X(X), XX-XX. Doi. xxx

### INTRODUCTION

Rapid advances in information technology have had a significant impact on various aspects of human life, including in terms of activity management and time management. In today's digital era, mobile devices such as smartphones are no longer only used to communicate, but also serve as versatile tools to access information and run various applications. One of the most widely used features is the reminder application, which plays a role in helping users record, schedule, and get notifications related to daily activities [1]. The app not only makes it easier for users to remember routine tasks, but also supports increased productivity through smart features such as automatic reminders, calendar integration, and notifications that can be customized to the user's preferences.

Mobile technology, especially the Android operating system, has become one of the most widely used platforms in Indonesia due to its open source, flexible, and wide user base [2]. Android provides a great opportunity for developers to create apps according to people's needs, including task reminder apps that support productivity and time use efficiency. Sabirin states that Android allows users to run, modify, and learn software for free, making it the right medium for developing educational applications [3]. This is in line with Veri Ilhadi's view that Android smartphones support various activities and transactions that demand speed and efficiency [4]. These two opinions show that Android is not only superior from a technical point of view, but also strategically as a medium for learning and activity management.

As a student, the ability to manage an activity schedule efficiently is very important to achieve academic success while maintaining a balance in daily life [5]. In the world of lectures/schools that are full of busyness, such as academic demands, part-time jobs, participation in organizations, and social activities, everything can feel burdensome if not managed properly [6]. Therefore, mastering time management skills is crucial so that the available time can be used optimally. Organized scheduling helps students avoid excessive stress, excessive fatigue, and procrastination habits that can have negative consequences. Through a priority list, students can time appropriately for each activity and reduce the possibility of procrastination [7]. In this context, digital reminder apps play a crucial role in maintaining students' consistency and focus on their academic responsibilities.

Along with the development of technology, various digital solutions have emerged to support time management, one of which is an Android-based task reminder app. This kind of app offers users the convenience of taking notes on tasks, setting automatic reminders, and monitoring the progress of completing tasks [8]. The Android system is a top choice due to its support for notification features, an easy-to-use interface, and its compatibility with various mobile devices [9]. With these advantages, Android-based task reminder apps are not only relevant for professionals, but also perfect for students and college students [10].

However, even though many reminder apps are available, the reality is that there are still many students who are not able to manage their time optimally. Based on the results of interviews with two students, both of them stated that they often forgot about assignment deadlines and schedules of academic activities. One of the respondents mentioned that he had difficulty managing his time due to the large number of tasks, while the other respondent admitted that he often missed deadlines because he did not have an automatically scheduled reminder system. These findings indicate a gap between the availability of reminder apps and the effectiveness of their use in everyday academic life [11].

Fatih Humam Ramadhan stated that the use of Android-based Mobile Learning applications can be an effective solution to improve students' understanding and abilities in various fields, including in terms of time management and assignment completion [12]. However, there are still a lack of applications that are specifically designed as academic reminders that focus on recording assignments, scheduling activities, and providing notifications that suit the needs of students. This shows that there is still room for development for applications that are more specific and adaptive to the needs of young users, especially in the context of education. The reminder feature integrated in the Mobile Learning application reinforces the dual function of this application as a learning tool as well as a management tool for daily academic activities [13].

The presence of this application, named Remindly, is expected to be able to improve time management, discipline, and user productivity in carrying out academic and other activities [15]. The app is designed to be a practical, easy-to-use, and intuitive digital assistant, with an automatic reminder feature to suit the needs of today's learners. The app's outputs include real-time notifications, daily to-do lists, and task completion statuses that help users stay on the productive path.

According to Muhammad Ridhan Maulana Nanda (2023), an application is needed that can be a solution to make it easier for students and students to manage and remember their entire activity schedule [14]. This statement reinforces the urgency of developing the Remindly app that not only serves as a reminder, but also supports the formation of more organized and responsible living habits. By taking advantage of the great potential of mobile devices and the flexibility of the Android system, this application makes an innovative contribution to the world of education as a learning aid as well as activity management.

This research aims to design and build an Android-based task reminder application aimed at helping users, especially students, in managing schedules and remembering various daily tasks

in a more systematic and organized manner. This application will include task recording features, setting deadlines, setting reminder times, and giving automatic notifications. Thus, this application is expected to provide concrete solutions to the problems of time management and student discipline that have been identified previously.

## **METHOD**

This research uses the Agile method in application development. Agile is an iterative and incremental approach that emphasizes cross-functional communication, infrastructure, continuous improvement, and rapid response to change [16]. This method was chosen because it is suitable for developing Remindly applications, which require adaptive processes, gradual testing, and continuous evaluation and improvement based on user feedback [17].

The development process involves multiple short cycles or sprints, which allows for constant assessment of project needs, planning adjustments, and development adaptations as per user feedback or priority changes. The results of the adjustments are summarized in the sprint backlog document, which serves as a working reference for each sprint cycle. In this study, three sprints are planned to gradually improve the features of the application. With a focus on regular delivery of functional products, intense team collaboration, and continuous interaction between developers and stakeholders, the Agile method aims to improve user satisfaction, development synchronization, and efficiency in achieving expected system targets [18].

The Agile Development methodology is in accordance with the SDLC (Software Development Life Cycle), including the planning, analysis, design, and coding phases. The Agile method is recognized to provide faster value, higher quality, and better predictability, especially in a dynamic and competitive mobile application development environment [19].

### **Research Subject**

The subjects of this study are students who have difficulty in organizing and remembering academic tasks. The selection of subjects was carried out by purposive sampling technique based on the characteristics and needs of Remindly application users. Initial data was obtained through interviews with two student resource persons:

1. Nurul Ilmi Syamsuddin: Students who have difficulty managing and remembering academic assignments. The online interview was conducted on Wednesday, February 26, 2025, at 19.30, with a duration of about 11 minutes.
2. Dessy Dwi Sulistiyawati: Students with similar difficulties. The online interview was held on the same day, at 19.57, with a duration of about 10 minutes.

The results of this interview are the basis for identifying user needs and the main problems that need to be addressed by the Remindly application. The selection of students as subjects is based on the fact that this group has a high intensity of academic tasks and requires practical solutions to manage and remember various tasks with different deadlines.

### **Stages of System Development**

#### **1. Planning System**

In Remindly's application planning, the Agile approach allows for flexible and step-by-step processes [20]. The demand for the system arises from the needs of students who often forget assignments and need a reminder app that is structured, easy to use, and accessible through Android devices. Feasibility studies show that the app can be developed economically and supported by a team that understands the needs of users.

#### **2. Analysis**

The analysis stage aims to understand and formulate the needs of users and systems as a whole. This is important to identify key issues in the development of the Remindly application, focusing on functional and non-functional needs [21]. Functional needs include core features such as the login system, task addition, reminder notifications, and user profile management. Non-functional needs include aspects of security, performance, reliability, and ease of use. This stage is visualized using UML diagrams, such as use case diagrams and activity diagrams.

### 3. Implementation

The implementation stage is the realization of the results of planning, analysis, and design that have been carried out. The implementation of the Remindly application is carried out in stages in three sprints according to the Agile methodology applied. The implementation phase is divided into two main parts:

#### a. Coding

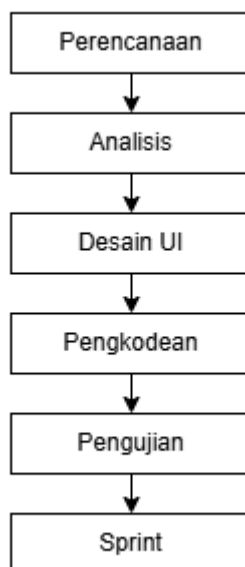
The system design is translated into program code using the appropriate framework and programming language. Development is divided into back-end and front-end. The back-end focuses on application logic and data management, while the front-end is responsible for the display of the user interface. The coding process follows an iterative approach, allowing for feature customization based on the evaluation of each sprint cycle [18].

#### b. Testing

Testing is done to ensure the Remindly app works properly and meets quality standards. The black box testing method is used, focusing on the behavior of the system from the user side without paying attention to the internal structure of the code. Testing focuses on inputs and outputs to ensure all key features perform as expected.

### Diagram of Agile Development Methods

Here is a diagram illustrating the Agile development methods used in this study:



**Figure 1.** Development Diagram

## RESULTS AND DISCUSSION

The result of development is in the form of a mobile task reminder application, Remindly, which is used by users, especially students, to organize and manage their daily tasks. Remindly's application development uses the Agile development method, which consists of four main stages, namely: system planning, needs analysis, design, and implementation which includes coding and testing. The presentation of the results of the Agile development stages for the Remindly application is as follows:

### Results of System Planning

The planning stage in the development of the Remindly application begins with conducting in-depth observations and interviews with potential users, namely students, to collect system needs. The results of this collection of needs are then documented in the form of a system request. This system request shows that there is a need for an Android-based platform that can help users manage, remind, and schedule their tasks easily and efficiently. The application is expected to increase productivity and user discipline in completing academic responsibilities. The following is Table 1 regarding the results of the planning system.

**Table 1.** Hasil Planning

<b>Android-Based Task Reminder Management System Feasibility Study – Remindly</b>
This feasibility study was conducted to support the development of an Android-based task reminder system, namely the Remindly application. The main focus of the app is to help students or other users manage their daily tasks effectively and on a schedule.
<b>Technical Feasibility Study:</b>
The development of the Remindly application is considered technically feasible. The development team is quite familiar with the technology used, namely Android Studio as the IDE and Java as the main programming language. The application does not rely on external servers or online databases such as MySQL or Firebase, as all task data is automatically stored in the local storage of the user's device. The technical risk is relatively low, but it is still necessary to pay attention to the compatibility of the application on various Android devices with at least version 7.0 Nougat and above, as well as optimization to keep local data storage secure and efficient.
<b>Economic Feasibility Study:</b>
From an economic point of view, the development of Remindly is relatively cost-effective because it uses open-source or free-to-use tools and services. This application has the potential to increase student productivity without having to spend a large amount of money, and opens up opportunities for the development of additional paid features in the future.
<b>Organizational Feasibility Study:</b>
Organizationally, this project has the full support of the development team and supervisors. The development of this application is in line with the need to improve the effectiveness of student time management, supporting the organization's mission to innovate in educational technology solutions.

Based on the results of the analysis of technical, economic, and organizational aspects, the development of the Remindly application is considered feasible to be realized. The estimated

completion of the project is estimated within 3 to 5 months, with the work carried out in a team consisting of 3 developers, namely Anggun Nurpadila, Artika Sari Dewi, and Zahwa Aurellia. Each team member has their own role in development, from planning, analysis, design, implementation, to application testing, so it is hoped that Remindly's development can run effectively and in accordance with the set targets.

### Analysis Results

At the analysis stage, the functional and non-functional needs of the application are defined in detail. Functional needs include key features such as survey creation, incentive distribution, and user management, while non-functional needs relate to data security, application performance, and usability which can be seen in detail in Table 2 and Table 3 below:

**Table 2.** Non-Functional Needs

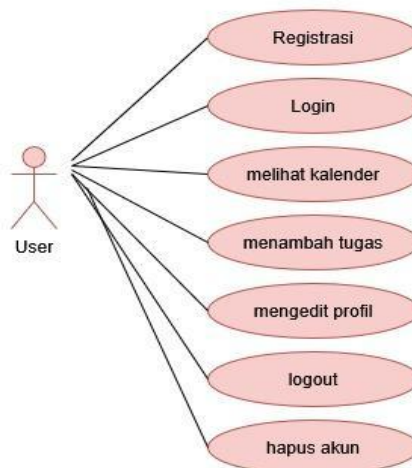
Non-Functional Requirement		
ID	Parameter	Explanation
NFR-01	Availability	Task reminders must remain active even when the app isn't open.
NFR-02	Ergonomy	The app should have a user-friendly look with easy-to-understand navigation
NFR-03	Security	Applications must protect user data from security attacks with secure authentication mechanisms.
NFR-04	Response Time	The splash screen must be displayed for 3 seconds before entering the main page.
NFR-05	Response Time	The user login process must be completed in less than 7 seconds.

**Table 3.** Functional Needs

Non-Functional Requirement		
ID	Parameter	Explanation
FR-01	Displaying the Splash Screen	This feature displays the initial view that appears when the app is first opened before entering the main page.
FR-02	Registering an Account	This feature Registers if the user does not have an account.
FR-03	Display	This feature allows users to enter their username and password if they already have an account.

FR-04	Displaying the Main Dashboard	This feature Presents the main page of the application after entering the username and password
FR-05	Provide Reminder Notifications	This feature Provides reminder notifications when a task is approaching a deadline.
FR-06	Allow users to add tasks	This feature allows users to add a list of tasks they want to do.
FR-07	Using Users to Set Task Deadlines	This feature allows users to set task collection deadlines.
FR-08	Organize tasks into 3 categories	This feature allows users to organize tasks into 3 categories (Undone, In Progress, Completed).
FR-09	Managing User Profiles	This feature allows users to view and edit user account information.

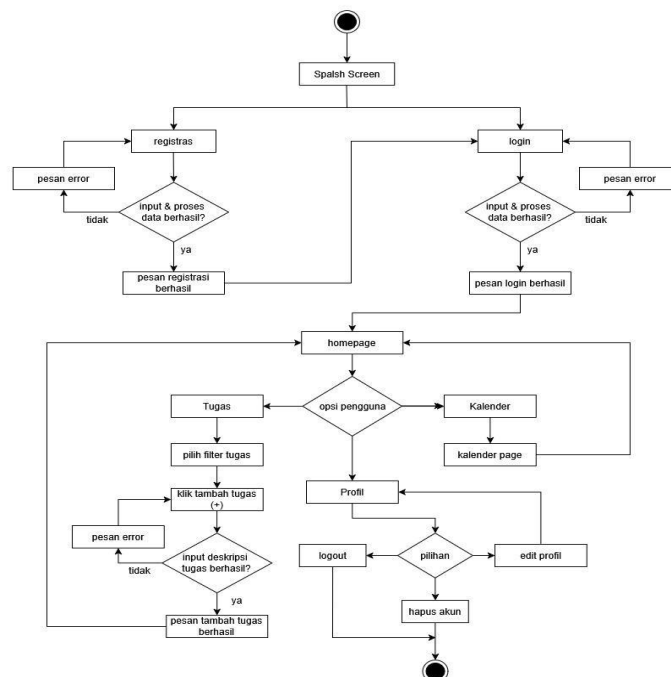
Apart from the definition of functional and non-functional needs, this analysis process is often represented through UML (Unified Modeling Language) diagrams, which allow for a clear visualization of the interaction between systems and users, which at this stage of analysis consists of use case diagrams that illustrate the various ways users interact with the system [22], and activity diagrams that show the workflows or processes in the system [23] ensuring that both types of these needs are effectively integrated to achieve the objectives of the system which can be seen in detail in Figure 2 and Figure 3:



**Figure 2.** Use Case Diagram

1. Use Case Registration This use case allows users to create a new account that can be used by filling in personal data such as name, username, email, and password.
2. Use Case Login: This use case allows users to log in and access the features in the application by using email and password parameters as authentication.

3. Use Case View Calendar: This use case allows users to view a calendar view that contains a list of tasks or schedules that have been added, making it easier for users to monitor task deadlines.
4. Use Case Adds Tasks: This use case allows users to add new tasks to the application by filling in information such as task name, description, deadline, and work status, and add a survey by paying an incentive through online payment
5. Use Case Edit Profile: This use case allows users to change their registered personal data such as name, email, or other account information stored in the application
6. Use Case Logout: This use case allows the user to log out of the account they are currently using, so there is no further access to personal data before logging back in.
7. Use Case Delete Account: This use case allows users to permanently delete accounts that have been registered, including all related data such as tasks and profile information.



**Figure 3.** Activity Diagram

## Design

The results of the design stage of this study are in the form of Class and Sequence Diagram & UI designs that have been adjusted to the references provided as well as the necessary adjustments according to the features of the incentive-based mobile survey application "Re-Quest".

### 1. Diagram Class Design

For visual design of programs, Unified Modeling Language (UML) is used as a framework or modeling language for object-oriented programming, UML modeling helps to simplify complex problems to make them easier to understand and learn. At this stage, modeling involves creating a class diagram that is used to define the class structure to be used in the application, including the essential attributes and methods, as well as the relationships between the classes which can be seen in the following Figure 4:



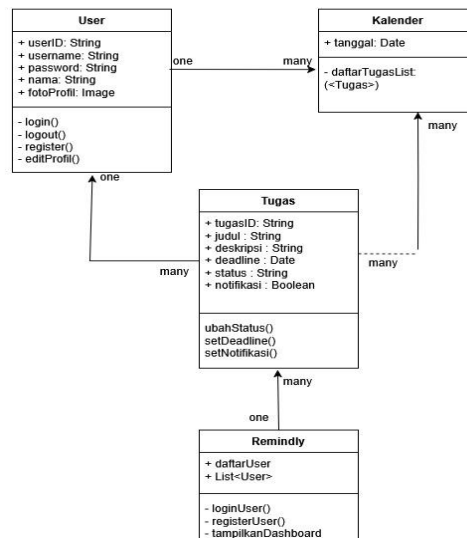


Figure 4. Class Diagram

and the sequence diagram used to explain the interactions between objects over time for a given process, provides a clear guide to the execution flow of operations in the application based on each use case [24] which can be seen in detail in Figure 5:

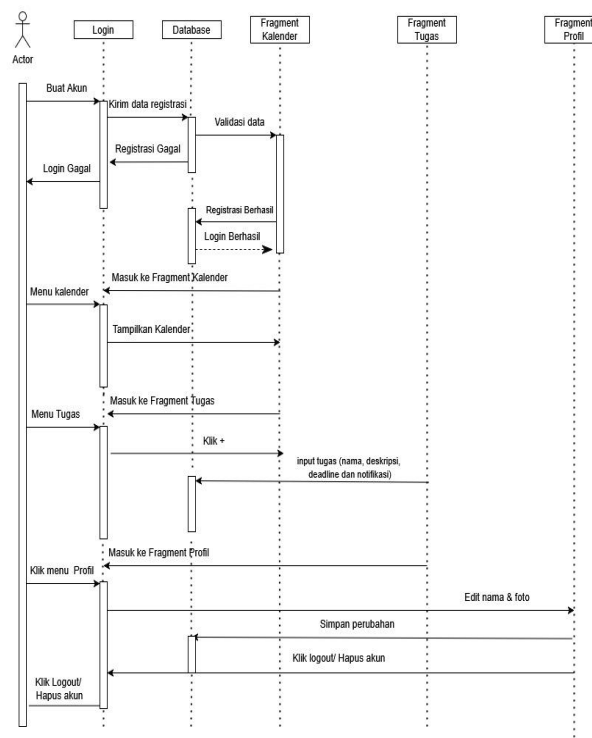
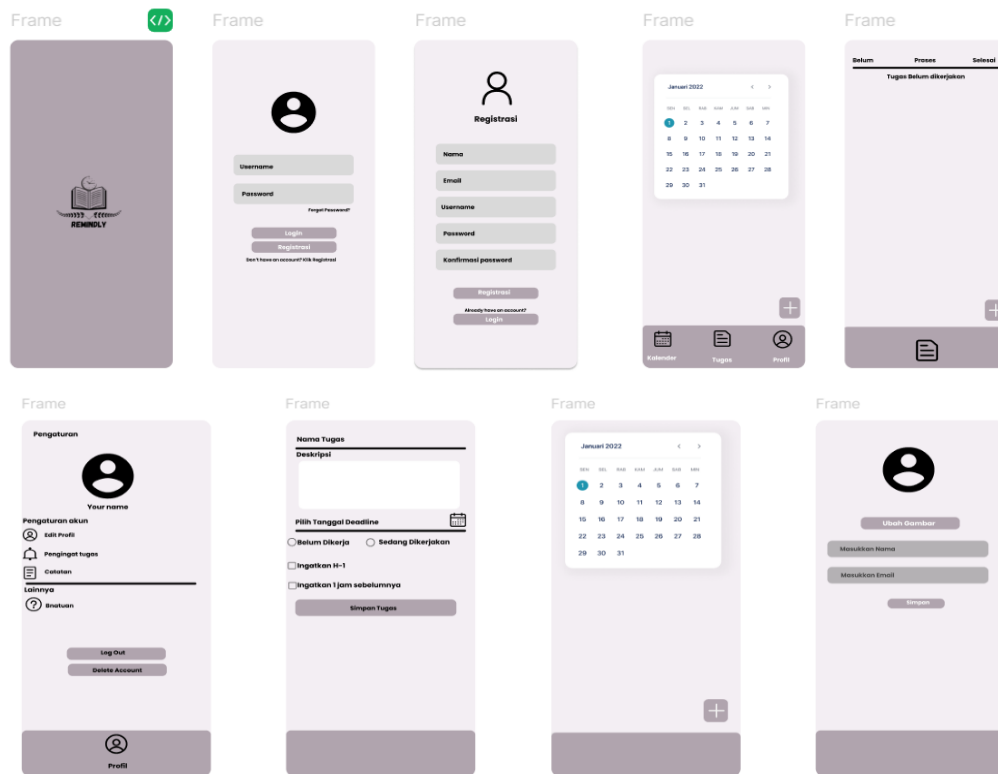


Figure 5. Sequence Diagram

## 2. Desain User Interface (UI)

Furthermore, the design of the Remindly application UI was carried out which was compiled based on the main functionalities that had been defined previously. The design process is carried out using the Figma design application, with reference to a special style guide to maintain visual consistency and make it easier for the development team in the user interface implementation process. Some of the icons and graphic elements in the design are taken from Google Icons and the free collection of Figma, which are free to use for non-commercial project purposes. The final UI design that has been approved includes the display of key features such as Splash Screen, Login, Dashboard, Add Tasks, Calendar, Notifications, Task Status, and User Profile. The visualization of

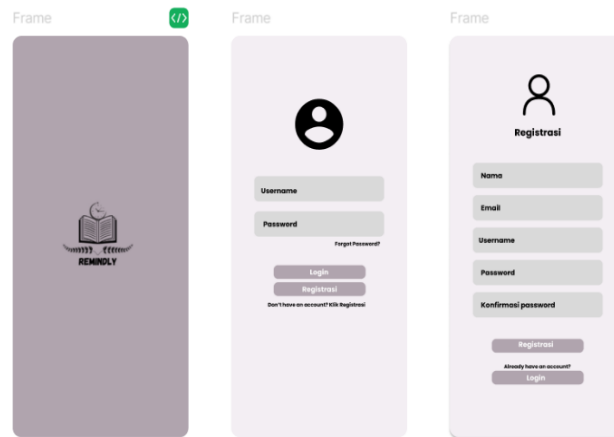
the UI design can be seen in the following Figure 6 as a representation of the Remindly application interface.



**Figure 6.** Style Guide

#### a. First Sprint

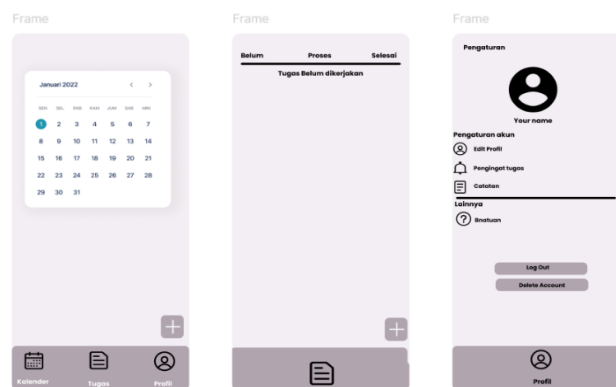
In the first sprint, a login or authentication page feature was developed, which consists of a login feature with email and password parameters, as well as a registration feature for new users through a simple form. Initially, the design only included these two features, but after a review and discussion by the team, the "Forgot Password" feature was added to the sprint backlog to make it easier for users to access their accounts if they forgot their password. Furthermore, the Splash Screen page was also developed as the initial display of the application when the user opens Remindly. After successfully logging in, users will be redirected to the main page that displays 5 main menus: Add Tasks, Task Calendar, Notifications, Task Status, and User Profile. This view is designed to allow users to instantly view and manage tasks efficiently, as shown in the following Figure 7:



**Figure 7.** Splash Screen Design, Login and Register Feature

b. Second Sprint

In the second sprint, two main features were developed, namely the main dashboard and menu navigation consisting of the Task Calendar, Task List, and User Profile. This feature allows users to choose a view based on their needs, whether they want to see tasks that need to be completed, check schedules through a calendar, or manage personal information on their profile. With this feature, users can directly access and manage their tasks from the main page, view schedules in a structured way on the calendar; and update personal data through their profiles. A view of these features can be seen in Figure 8, which shows the interface design for the Dashboard, Calendar, and User Profile.

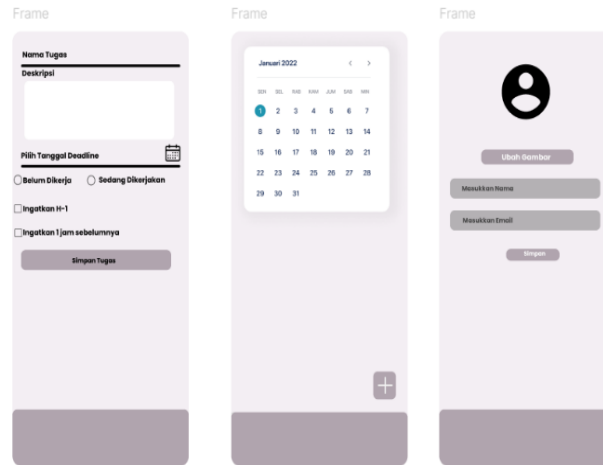


**Figure 8.** Design Calendar Features, tasks and profiles

c. Third Sprint

In the third sprint, development is focused on two main features, namely the Calendar and Add Tasks pages, as well as the User Profile edit feature. On the Calendar page, users can view all scheduled tasks in a structured manner by date, making it easier for them to plan and manage the timing of their tasks. Users can also navigate to specific dates to view or review tasks adjacent to the deadline. The Add Task feature allows users to add new tasks by filling in details such as task name, deadline, description, and task status. Once a task is added, it will automatically appear on the Calendar and Task Status page, making it easier for users to monitor their progress. Furthermore, the User Profile edit feature was also developed, where users can update basic information such as names

and profile photos, but cannot change registered emails. This feature is designed to provide flexibility in account management, without compromising the user's key data. The full look and functionality of this sprint can be seen in Figure 9, which shows the Calendar, Add Tasks, and User Profile page interfaces



**Figure 9.** Page Design Features Calendar, tasks and profiles

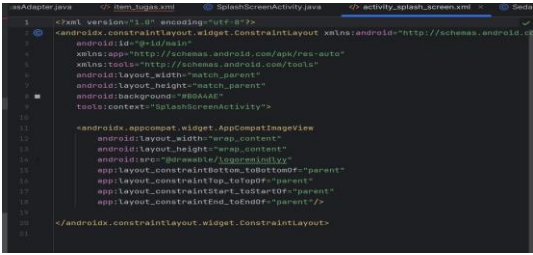
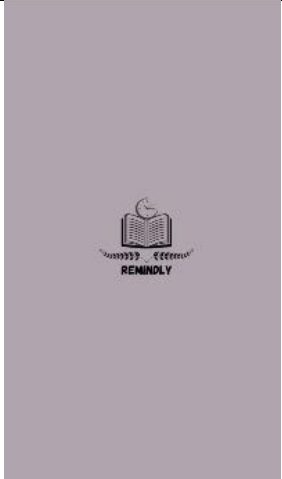
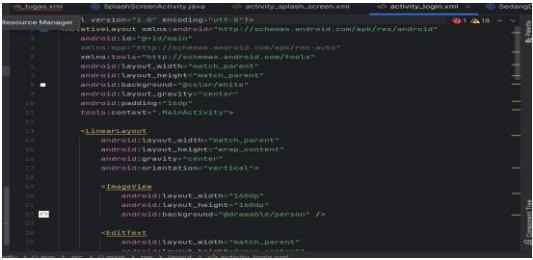

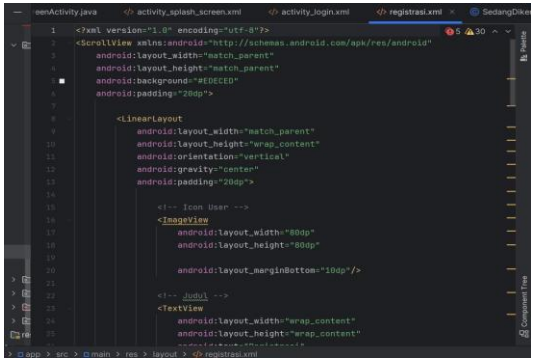
#### d. Implementation Results

Based on the results of the analysis and design that has been carried out, several changes have been obtained which are grouped into three sprints and have been implemented through the coding process using Android Studio. After going through the testing stage, an Android-based task reminder application called "Remindly" was obtained.

#### e. Coding Results

The result of the coding stage in this project is the implementation of the Remindly application using Android Studio as the main tool. Android Studio was chosen because it supports the Java language and provides full features for the development process, including debugging and emulators for testing. The coding is done based on the UI design that was pre-designed in Figma. To store and manage data in real-time, Firebase is used as a backend, as it is easy to integrate and has good security features. All functionality is tested using the Android Studio emulator to ensure that the app runs properly before being used by the user. This includes testing user interaction with the survey system ensuring that all aspects of the application work as expected prior to market launch, the results of which can be seen in the following Table 4:

**Table 4.** Coding and Results View

Coding	Results Display
First Sprint	
Splash Screen, Login, dan Register	
 <p><b>Figure 10.</b> Splash Screen Code Display</p>	 <p><b>Figure 13.</b> Splash Screen Page App View</p>
 <p><b>Figure 11.</b> Login Page Code Display</p>	 <p><b>Figure 14.</b> Login Page App View</p>
 <p><b>Figure 12.</b> Register Page Code Display</p>	<p><b>Figure 13.</b> Register Page App View</p>

**Registrasi**

Nama

Email

Username

Password

Konfirmasi Password

**Registrasi**

Already have an account?

**Login**

Figure 15. Register Page Application Display

## Home, Calendar, Tasks, and Profile

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     android:layout_width="match_parent"
5     android:layout_height="match_parent"
6     android:orientation="vertical">
7
8     <!-- Container untuk Fragment -->
9     <FrameLayout
10         android:id="@+id/fragment"
11         android:layout_width="match_parent"
12         android:layout_height="500dp"
13         android:layout_weight="1"/>
14
15     <!-- Bottom Navigation -->
16     <com.google.android.material.bottomnavigation.BottomNavigationView
17         android:id="@+id/bottomNavigationView"
18         android:layout_width="match_parent"
19         android:layout_height="wrap_content"
20         app:menu="@menu/nav_bottom_menu" />
21 </LinearLayout>
```

Figure 16. Home Page Code Display



Figure 20. Home Page App View

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     android:layout_width="match_parent"
5     android:layout_height="match_parent">
6
7     <!-- Widget CalendarView dengan constraints -->
8     <CalendarView
9         android:id="@+id/calendarView"
10         android:layout_width="match_parent"
11         android:layout_height="wrap_content"
12         android:layout_marginTop="20dp"
13         app:layout_constraintTop_toTopOf="parent"
14         app:layout_constraintStart_toStartOf="parent"
15         app:layout_constraintEnd_toEndOf="parent"/>
16 </androidx.constraintlayout.widget.ConstraintLayout>
```

Figure 17. Calendar Page Code Display

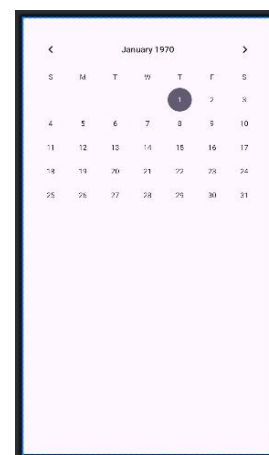


Figure 21. Calendar Page App View

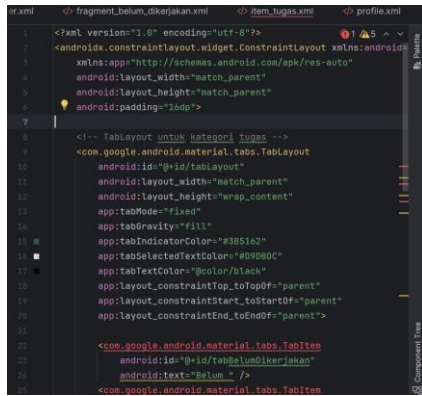


Figure 18. Task Page Code Display



Figure 22. Task Page App View

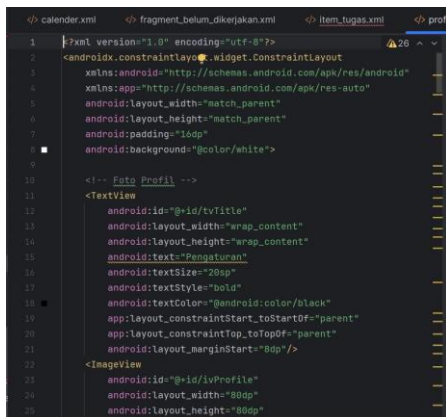


Figure 19. Profile Page Code Display

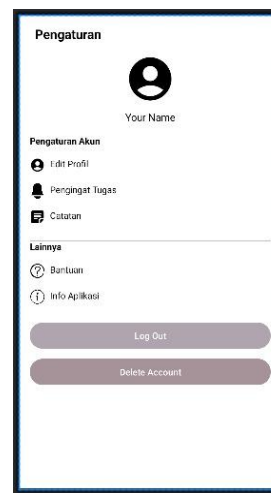


Figure 23. Profile Page App View

#### f. White Box Test Results

Application performance testing is carried out to determine the effectiveness of the developed application. Testing is conducted using a white-box that focuses on the functionality side, specifically on the input and output system testing [25], white-box is a software testing method that is carried out by directly examining the structure and logic of the program's code. The goal is to ensure that every track in the program has been tested and runs as expected without any logical errors. Here's a white-box test for the Registration-Login, Add Task, Move Room Task:

##### 1. Registration

```
package com.example.remindly;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
```

```

public class MainActivity extends AppCompatActivity {

    3 usages
    EditText etUsername, etPassword;
    2 usages
    Button btnLogin, btnRegister;
    2 usages
    SharedPreferences sharedPreferences;
    2 usages
    private DatabaseReference database;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        etUsername = findViewById(R.id.inputUserName);
        etPassword = findViewById(R.id.inputPassword);
        btnLogin = findViewById(R.id.btn_masuk);
        btnRegister = findViewById(R.id.btnRegister);

        sharedPreferences = getSharedPreferences("UserData", MODE_PRIVATE);
        database = FirebaseDatabase.getInstance().getReferenceFromUrl("https://remindly-8f151-default-rtdb.firebaseio.com/");

        btnLogin.setOnClickListener(new View.OnClickListener() {
            String usernameInput = etUsername.getText().toString().trim();
            String passwordInput = etPassword.getText().toString().trim();
            if (usernameInput.isEmpty()) {
                etUsername.setError("Username tidak boleh kosong");
                return;
            }
            if (passwordInput.isEmpty()) {
                etPassword.setError("Password tidak boleh kosong");
                return;
            }

            // Cek user di Firebase
            database.child(pathString: "users").child(usernameInput).addListenerForSingleValueEvent(new ValueEventListener() {
                2 usages
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {
                    if (dataSnapshot.exists()) {
                        // User ditemukan
                        String firebasePassword = dataSnapshot.child(path: "password").getValue(String.class);

                        // Bandingkan password langsung (karena di Firebase disimpan plain text)
                        if (passwordInput.equals(firebasePassword)) {
                            // Login sukses
                            Toast.makeText(context: MainActivity.this, text: "Login Berhasil!", Toast.LENGTH_SHORT).show();

                            // Simpan data user ke SharedPreferences
                            SharedPreferences.Editor editor = sharedPreferences.edit();
                            editor.putString("username", usernameInput);
                            editor.putString("nama", dataSnapshot.child(path: "nama").getValue(String.class));
                            editor.putString("email", dataSnapshot.child(path: "email").getValue(String.class));
                            editor.putBoolean("isLoggedIn", true);
                            editor.apply();

                            // Pindah ke Dashboard
                            Intent intent = new Intent(packageContext: MainActivity.this, DashboardActivity.class);
                            startActivity(intent);
                            finish();
                        } else {
                            Toast.makeText(context: MainActivity.this, text: "Password salah!", Toast.LENGTH_SHORT).show();
                        }
                    } else {
                        Toast.makeText(context: MainActivity.this, text: "User tidak ditemukan!", Toast.LENGTH_SHORT).show();
                    }
                }
            });

            @Override
            public void onCancelled(DatabaseError databaseError) {
                Toast.makeText(context: MainActivity.this, text: "Database error: " + databaseError.getMessage(), Toast.LENGTH_SHORT).show();
            }
        });

        btnRegister.setOnClickListener(new View.OnClickListener() {
            Intent intent = new Intent(packageContext: MainActivity.this, RegistrasiActivity.class);
            startActivity(intent);
        });

        // Fungsi hash tetap dipertahankan untuk kebutuhan lain
        no usages
        private String hashPassword(String password) {
            try {
                MessageDigest digest = MessageDigest.getInstance(algorithm: "SHA-256");
                byte[] hash = digest.digest(password.getBytes());
                StringBuilders hexString = new StringBuilders();
                for (byte b : hash) {
                    String hex = Integer.toHexString(0xff & b);
                    if (hex.length() == 1) hexString.append('0');
                    hexString.append(hex);
                }
                return hexString.toString();
            } catch (NoSuchAlgorithmException e) {
                e.printStackTrace();
                return "";
            }
        }
    }
}

```

Figure 24. Registration-Login Coding



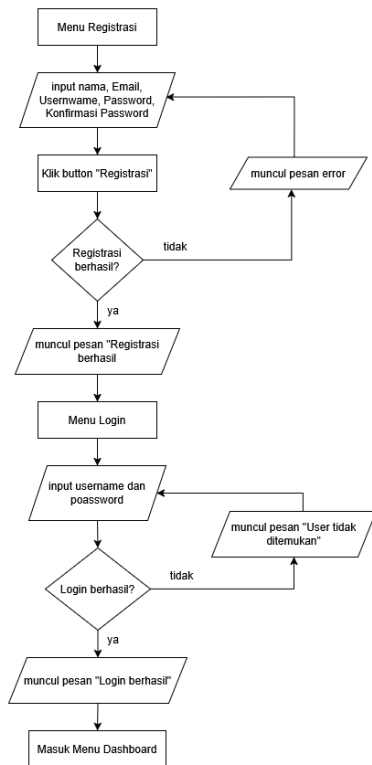


Figure 25. Registration Flowchart

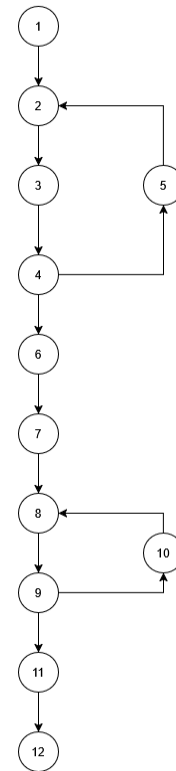


Figure 26. Flowgraph Registrasi

$$V(G)=E-N+2$$

E = Number of Edges (logical arrows/arrows connecting between nodes)

N = Number of Nodes (decision points, processes, inputs/outputs in flowchart/flowgraph)

$$V(G)=E-N+2= 13 - 12 + 2 = 3$$

This means that there are 3 independent paths that need to be tested.

Table 5. White Box Registration-Login Test Results

Path	1
Line	1-2-3-4-6-7-8-9-11-12
Skenario	1. Registration Menu 2. Input Data 3. Click the Registration button 4. Data Validation (Valid) 6. "Registration Successful" message appears 7. Menu Login 8. Input Username & Password 9. Data validation 11. "Login Successful" Message Appears 12. Enter the dashboard menu

Test Results	Succeed
<b>Path</b>	<b>2</b>
Line	1-2-3-4-5-2
Skenario	1. Registration Menu2. Data Input3. Click on the Registration button for Task item 4. Data Validation (Invalid) 5. Error message appears 2. Input Data
Test Results	Succeed
<b>Path</b>	<b>3</b>
Line	1-2-3-4-5-2-3-4-6-7-8-9-10-8-9-11-12
Skenario	1. Registration Menu2. Data Input3. Click the Registration button4. Data Validation (Invalid) 5. Error message appears 2. Data Input3. Click the Registration button4. Data Validation (Valid) 6. A "Registration Successful" message appears 7. Menu Login 8. Username & Password Input9. Data validation (Invalid) 10. "User not found" message appears 8. Username & Password Input9. Data validation (Valid) 11. "Login Successful" Message Appears 12. Enter the dashboard menu
Test Results	Succeed

After the number of independent pathways is known, a comparison will be made using the table of the relationship between cyclomatic complexity and risk in the following Table 6:

**Table 6.** Risks of the Application Procedure Level

Value V(G)	Procedure Type	Risk Level
1-4	Simple procedure	Low
5-10	Structured and stable procedures	Low to medium
11-20	More complex procedures	Winning
21-50	Complex and critical rosedur	Medium-High
>50	Very complex procedures, error-prone	Very High

The results of the cyclomatic complexity calculation show that the Registration-Login feature is classified as a simple procedure with low risk, because it only has three independent logic paths. After the path is determined, the next stage is to conduct testing based on the test cases presented in the following Table 7:

**Table 7.** Test Case Adds Task

Yes	Scenario Name	Input	Expected Results	Result	Result
1	Login Successfully	Correct Email & Password	Go to the dashboard menu	The user was successfully redirected to the main view after logging in	Valid
2	Registration failed	Invalid data, not registered	Registration error message, keep the registration page	The system rejects the registration and the user remains in the registration form	Valid
3	Login after registration and login failed	Invalid data continue to fill in registration data, Email/Password is wrong, login again	Successfully Register and Login, Enter the Dashboard menu	The account was successfully created and the user went directly to the main page of the application	Valid

## 2. Add Tasks

```

package com.example.remindly;

import android.app.DatePickerDialog;
import android.content.ContentValues;
import android.net.Uri;
import android.os.Bundle;
import android.provider.CalendarContract;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

import java.util.Calendar;

public class TambahTugasActivity extends AppCompatActivity {

    3 usages
    EditText etNamaTugas, etDeskripsi, etDeadline;
    2 usages
    Button btnSimpan, btnTambahKeKalender;
    2 usages
    RadioGroup radioGroupStatus;
    7 usages
    Calendar calendar;
    5 usages
    String deadline;
    3 usages
    DatabaseReference tugasRef;

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tambah_tugas);

    etNamaTugas = findViewById(R.id.etNamaTugas);
    etDeskripsi = findViewById(R.id.etDeskripsi);
    etDeadline = findViewById(R.id.etDeadline);
    btnSimpan = findViewById(R.id.btnSimpan);
    btnTambahKekalender = findViewById(R.id.btnAddKekalender);
    radioGroupStatus = findViewById(R.id.radioGroupStatus); // pastikan ID sesuai

    calendar = Calendar.getInstance();
    tugasRef = FirebaseDatabase.getInstance().getReference().child("tugas");
    etDeadline.setOnClickListener(new View.OnClickListener() {
        DatePickerDialog datePickerDialog = new DatePickerDialog(
            context, new DatePickerDialog.OnDateSetListener() {
                @Override
                public void onDateSet(DatePicker view, int year, int month, int dayOfMonth) {
                    calendar.set(year, month, dayOfMonth, hourOfDay: 8, minute: 0);
                    deadline = dayOfMonth + "/" + (month + 1) + "/" + year;
                    etDeadline.setText(deadline);
                }
            }, calendar, calendar.get(Calendar.YEAR),
            calendar.get(Calendar.MONTH),
            calendar.get(Calendar.DAY_OF_MONTH));
        datePickerDialog.show();
    });

    btnSimpan.setOnClickListener(new View.OnClickListener() {
        String namaTugas = etNamaTugas.getText().toString().trim();
        String deskripsi = etDeskripsi.getText().toString().trim();
        int selectedId = radioGroupStatus.getCheckedRadioButtonId();

        if (namaTugas.isEmpty() || deskripsi.isEmpty() || deadline == null || selectedId == -1) {
            Toast.makeText(context, this, "Harap isi semua data!", Toast.LENGTH_SHORT).show();
            return;
        }

        RadioButton selectedStatus = findViewById(selectedId);
        String status = selectedStatus.getText().toString(); // Misal: "Belum", "Sedang", "Selesai"

        // Buat ID tugas
        String id = tugasRef.push().getKey();
        Tugas tugas = new Tugas(id, namaTugas, deskripsi, deadline, status);

        // Simpan tugas ke Firebase
        if (id != null) {
            tugasRef.child(id).setValue(tugas); // Menyimpan data tugas dengan ID unik
            Toast.makeText(context, this, "Tugas berhasil ditambahkan!", Toast.LENGTH_SHORT).show();
            finish(); // Tutup activity setelah tugas disimpan
        } else {
            Toast.makeText(context, this, "Gagal menambahkan tugas!", Toast.LENGTH_SHORT).show();
        }
    });

    btnTambahKekalender.setOnClickListener(new View.OnClickListener() {
        String namaTugas = etNamaTugas.getText().toString().trim();
        String deskripsi = etDeskripsi.getText().toString().trim();

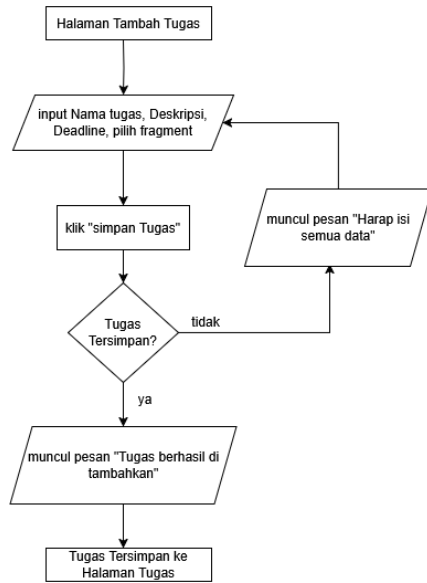
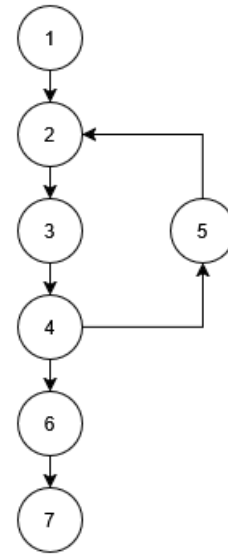
        if (namaTugas.isEmpty() || deskripsi.isEmpty() || deadline == null) {
            Toast.makeText(context, this, "Harap isi semua data!", Toast.LENGTH_SHORT).show();
        } else {
            tambahKekalender(namaTugas, deskripsi);
        }
    });
}

// Usage
private void tambahKekalender(String namaTugas, String deskripsi) {
    ContentValues eventValues = new ContentValues();
    eventValues.put(CalendarContract.Events.CALENDAR_ID, 1);
    eventValues.put(CalendarContract.Events.TITLE, namaTugas);
    eventValues.put(CalendarContract.Events.DESCRPTION, deskripsi);
    eventValues.put(CalendarContract.Events.EVENT_LOCATION, "Remindly App");
    eventValues.put(CalendarContract.Events.DTSTART, calendar.getTimeInMillis());
    eventValues.put(CalendarContract.Events.DTEND, calendar.getTimeInMillis() + 60 * 60 * 1000); // Durasi 1 jam
    eventValues.put(CalendarContract.Events.EVENT_TIMEZONE, calendar.getInstance().getTimeZone().getID());

    Uri uri = getContentResolver().insert(CalendarContract.Events.CONTENT_URI, eventValues);
    if (uri != null) {
        Toast.makeText(context, this, "Tugas ditambahkan ke kalender!", Toast.LENGTH_SHORT).show();
    } else {
        Toast.makeText(context, this, "Gagal menambahkan tugas ke kalender!", Toast.LENGTH_SHORT).show();
    }
}

```

Figure 27.Add Task Coding

**Figure 28.** Flowchart Add Task**Figure 29.** Flowgraph Add Task

$$V(G)=E-N+2$$

E = Number of Edges (logical arrows/arrows connecting between nodes)

N = Number of Nodes (decision points, processes, inputs/outputs in flowchart/flowgraph)

$$\text{Cyclomatic Complexity } (V(G)) = E - N + 2 = 8 - 7 + 2 = 3$$

That is: there are 3 independent paths that must be tested.

**Table 8.** White Box Testing Results Add Task Feature

Path	1
Line	1-2-3-4-5-6-7
Skenario	1. Add Task Page 2. Data Input3. Click the Task Simoan button4. Input validation (valid) 5. Show "Task Successfully Added" message 6. Tasks Saved to the Task Page
Test Results	Succeed
Path	2
Line	1-2-3-4-5-2
Skenario	1. Add Task Page 2. Data Input3. Click the Simoan Task 4 button . Input validation (invalid) 5. Show the message "Please fill in all data" 2. Data Input Back
Test Results	Succeed
Path	3
Line	1-2-3-4-5-2-3-4-5-6-7

Skenario	1. Add Task Page 2. Data Input3. Click the Task Simoan button4. Input validation (invalid) 5. Show the message "Please fill in all data" 2. Data Input Back  3. Click the Task Summary4 button. Input validation (valid) 5. Show "Task Successfully Added" message 6. Tasks Saved to the Task Page
Test Results	Succeed

Once the number of independent pathways is known, a comparison is then made using a table showing the relationship between cyclomatic complexity and risk level, as shown in the following Table 9:

**Table 9.** Risks of the Application Procedure Level

Value V(G)	Procedure Type	Risk Level
1-4	Simple procedure	Low
5-10	Structured and stable procedures	Low to medium
11-20	More complex procedures	Winning
21-50	Complex and critical rosedur	Medium-High
>50	Very complex procedures, error-prone	Very High

Based on the results of the cyclomatic complexity calculation, the Add Task feature is categorized as a simple procedure with a low level of risk, as it has only three independent logic paths tested. Then, after knowing the independent path, the next step is a test case, as shown in Table 10 below:

**Table 10.** Test Case Adds Task

Yes	Scenario Name	Input	Expected Results	Result	Result
1	Add Valid Assignments	All fields are filled in correctly	Task saved, the message "Task successfully added" appears	The data was successfully saved and the message "Task successfully added appeared" on the screen	Valid
2	Add Blank Tasks	There is a blank column (blank title/date)	Pops up "Please fill in all data" message, doesn't save the task	The app displays the message "Please fill in all data" and the task is not saved	Valid

3	Add Repeat Tasks	Initially the input was empty, then fixed	Initially errors, then succeed when the input is repaired and saved again	After the input improvement, the storage process was smooth	Valid
---	------------------	---	---	---	-------

### 3. Moving Duties

```

package com.example.remindly;

import androidx.annotation.NonNull;
import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;
import androidx.lifecycle.ViewModel;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

import java.util.ArrayList;
import java.util.List;

10 usages
public class TugasViewModel extends ViewModel {

    4 usages
    private final MutableLiveData<List<Tugas>> tugasBelum = new MutableLiveData<> (new ArrayList<>());
    4 usages
    private final MutableLiveData<List<Tugas>> tugasProses = new MutableLiveData<> (new ArrayList<>());
    4 usages
    private final MutableLiveData<List<Tugas>> tugasSelesai = new MutableLiveData<> (new ArrayList<>());

    no usages
    public TugasViewModel() {
        // Ini penting agar data langsung diambil dari Firebase saat ViewModel dibuat
        muatSemuaTugas();
    }

    no usages
    public LiveData<List<Tugas>> getTugasBelum() { return tugasBelum; }

    1 usage
    public LiveData<List<Tugas>> getTugasProses() { return tugasProses; }

    1 usage
    public LiveData<List<Tugas>> getTugasSelesai() { return tugasSelesai; }

    1 usage
    public void tambahTugas(Tugas tugas) {
        List<Tugas> list = getListByStatus(tugas.getStatus());
        if (list != null) {
            list.add(tugas);
            updateList(tugas.getStatus(), list);
        }
    }

    3 usages
    public void pindahStatus(Tugas tugas, String dariStatus, String keStatus) {
        DatabaseReference tugasRef = FirebaseDatabase.getInstance().getReference().child("tugas").child(tugas.getId());
        tugasRef.child("status").setValue(keStatus);

        List<Tugas> dariList = getListByStatus(dariStatus);
        List<Tugas> keList = getListByStatus(keStatus);

        if (dariList != null && keList != null) {
            dariList.remove(tugas);
            tugas.setStatus(keStatus);
            keList.add(tugas);
            updateList(dariStatus, dariList);
            updateList(keStatus, keList);
        }
    }

    2 usages
    public void hapusTugas(Tugas tugas, String status) {
        List<Tugas> list = getListByStatus(status);
        if (list != null) {
            list.remove(tugas);
            updateList(status, list);
        }

        FirebaseDatabase.getInstance().getReference().child("tugas").child(tugas.getId()).removeValue();
    }

    public void muatSemuaTugas() {
        DatabaseReference tugasRef = FirebaseDatabase.getInstance().getReference().child("tugas");
        tugasRef.addValueEventListener(new ValueEventListener() {

            2 usages
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                List<Tugas> listBelum = new ArrayList<>();
                List<Tugas> listProses = new ArrayList<>();
                List<Tugas> listSelesai = new ArrayList<>();
                for (DataSnapshot data : snapshot.getChildren()) {

```

### Figure 30. Coding Switching Tasks



This means that there are 2 independent paths that need to be tested.

**Table 11.** White Box Testing Results Add Task Feature



<b>Path</b>	<b>1</b>
Line	1-2-3-4-5-8-9
Skenario	1. Assignment Page 2. Select Tasks3. Click on the Task item 4. Select the assignment room5. Select "Move Task to Not Yet"
Test Results	8. Task of moving room9. Tasks Saved on the Task Page Succeed
<b>Path</b>	<b>2</b>
Line	1-2-3-4-6-8-9
Skenario	1. Assignment Page 2. Select Tasks3. Click on the Task item 4. Select the task room6. Select "Move Task to Process"
Test Results	8. Task of moving room9. Tasks Saved on the Task Page Succeed
<b>Path</b>	<b>3</b>
Line	1-2-3-4-7-8-9
Skenario	1. Assignment Page 2. Select Tasks3. Click on the Task item 4. Select the task room7. Select "Move Task to Done"
Test Results	8. Task of moving room9. Tasks Saved on the Task Page Succeed

Once the number of independent pathways has been obtained, the next step is to compare them using a table illustrating the relationship between cyclomatic complexity and risk levels, as presented in Table 12 below:

**Table 12.** Risks of the Application Procedure Level

<b>Value V(G)</b>	<b>Procedure Type</b>	<b>Risk Level</b>
1-4	Simple procedure	Low
5-10	Structured and stable procedures	Low to medium
11-20	More complex procedures	Winning
21-50	Complex and critical rosedur	Medium-High
>50	Very complex procedures, error-prone	Very High

Based on cyclomatic complexity calculations, the Task Switching feature falls into the category of simple procedures with a low level of risk, as it consists of only three independent logic paths. Once the pathway is identified, the next step is to test using the test case as shown in Table 13 below:

**Table 13.** Test Case for Job Switching

Yes	Scenario Name	Input	Expected Results	Result	Result
1	Move to Not Yet	Select the task, select "Move to Not Yet"	Task to move to room "Yet"	Task items are successfully displayed in the undone task category	Valid
2	Moving to Process	Select the task, select "Move to Process"	Task moves to the "Process" room	Tasks appear in the section indicating in progress	Valid
3	Move to Done	Select a task, select "Move to Done"	Task moves to the "Done" room	Task successfully redirected to the completed task list	Valid

g. Black Box Test Results

The test is done using the Black Box method where the user tests the functionality without knowing the program's logic. The goal of this test is to ensure that every key feature of the Remindly app works as expected. The test results are shown in the following Table 14:

**Table 14.** Test Case for Job Switching

Black Box Testing					
Test Code	Test Case	Expected Results	Results Obtained	Status	Test Code
UB-1	Log in with a blank username and password	The error message "Username and password must not be blank" is displayed	"Username and password should not be blank" error message	Succeed	UB-1
UB-2	Log in with the wrong username and password	"Login Failed" error message	The error message "User not found!" "Appearance"	Succeed	UB-2
UB-3	Log in with the correct username and password	Successful login and redirect to the dashboard	Successfully log in and redirect to the dashboard	Succeed	UB-3

UB-4	Register with complete data	"Successful registration" message and redirect to dashboard	Message "Registration successful" and redirect to the dashboard	Succeed	UB-4
UB-5	Register without filling in any of the indicators	Error message "Indicator must be filled" displayed	The error message "Indicator must be filled" appears	Succeed	UB-5
UB-6	Register with a password of less than 6 characters	The error message "Password must be 6 digits" is displayed	The error message "Password must be 6 digits" appears	Succeed	UB-6
UB-7	Add an untitled/dated task	A "Title and date required" error message is displayed	Error message "Title and date required" is displayed	Succeed	UB-7
UB-8	Add reminders to tasks	Reminder notifications set up successfully	Reminder notification successfully displayed	Succeed	UB-8
UB-9	Edit a task you've created	Task updated successfully	Task updated successfully	Succeed	UB-9
UB-10	Remove a task from the list	Tasks deleted and don't appear in the task list	Tasks deleted and not appearing	Succeed	UB-10
UB-11	Access and change profile data	Profile data is successfully updated	Profile data is successfully updated	Succeed	UB-11
UB-12	Log out of the app	The user returns to the login page	The user is redirected to the login page	Succeed	UB-12

The Remindly app is compared to some previous studies to assess its advantages and disadvantages:

**Table 15.** Previous Research

Research	Key Features	Platform	Excess	Deficiency
Rahman et al. (2020)	Reminder tugas harian	Android	Simple, lightweight display	No automatic reminders

---

Sari & Nugroho (2021)	Calendar and reminders	Web	Integrated calendar	Can't go offline
Rainfall (2022)	Learning reminders	Android	Learning focus	No status tracking
Squirting (2023)	Student time management	Android	Priority system	Doesn't support time notifications
This research	Task reminders, notifications, task status	Android	Full-featured, intuitive, real-time	Can't go offline

---

The results of the comparison show that Remindly has more complete features tailored to the needs of students, including task status tracking and time-based automatic reminders. The app addresses some of the shortcomings of previous apps that didn't support automatic reminders or didn't provide grouping of tasks by status. However, Remindly can still be further developed, for example with the addition of the Cannot Be Offline feature so that user data can be accessed from various devices. Development can also include location-based reminders and external calendar integration. The Agile approach to development provides an advantage in flexibility and responsiveness to user feedback.

Based on the development results, there are several key features that have been successfully implemented and tested in the Remindly application, with testing carried out using the black-box testing method. The first feature developed was the login feature, which serves as authentication to keep user data safe, using email and password as the main parameters. Security is very important because this app deals with users' personal data. Next, there is a registration feature, which allows new users to create an account by filling in information such as name, email, and password. This registration process is equipped with input validation so that users fill in all mandatory data. To anticipate user problems in accessing accounts, a forgot password feature was also developed, which allows users to reset passwords through email verification if they forgot their previous password. Validation messages such as "Password must be 6 digits" are displayed to ensure the security and consistency of the input format.

Apart from authentication, the main feature of this app is task management. Users can add a complete task with a title, description, and date. If the data is incomplete, the system will display an error message to remind the user. The app also allows users to add reminders to each task they create. This reminder is integrated with device notifications, so users will receive notifications at a predetermined time. Users can edit or update tasks, as well as delete tasks that are no longer needed. All changes are saved and instantly reflected in the to-do list. Another feature available is profile management, where users can access and change profile data such as names and passwords, except for fixed emails.

Finally, there is a logout feature that returns users to the login page to maintain the privacy and security of the session.

## CONCLUSIONS

The Remindly app was successfully implemented using the Agile method as a task reminder solution that helps users manage daily activities efficiently. Features such as login, registration, task reminders, edit and delete tasks, and notifications work well and have been tested using the Black Box method, showing results that meet expectations and supporting an intuitive and effective user experience. The contribution of this research lies in the integration of task category-based reminders and work status that have not been widely applied in similar studies. Additionally, it is important to state the limitations of the study, such as aspects that have not been explored or variables that may have influenced the results, to provide a more comprehensive picture of the scope of this study.

As a suggestion, further development could include additional features such as calendar sync, task completion statistics, as well as location-based or priority reminders to improve the functionality of the app. Further long-term research is also recommended to evaluate the impact of the app on user productivity and consistency in completing daily tasks.

## REFERENCES

- [1] E. R. Syahputra, "Design and Build a Package Status Reminder Information System Based on SMS Gateway in Logistics," *Sem. Nas. Technology. Common Inf.*, vol. 1, no. 1, pp. 364–371, 2021.
- [2] G. Primadana Edde and K. Budayawan, "Creation of Lecture Schedule Reminder Application in the Department of Electronics Engineering Based on Android," *Voteteknika (Vocational Tek. Elektron. dan Inform.*, vol. 9, no. 4, p. 1, 2021, doi: 10.24036/voteteknika.v9i4.112669.
- [3] A. R. Sabirin and I. K. D. Mardikayasa, "309-1456-1-Pb," vol. 10, no. 1, pp. 19–25, 2021.
- [4] V. Ilhadi, D. Ardiansyah, and M. Muthmainnah, "Mobile-Based Activity Schedule Reminder Application," *Sisfo J. Ilm. Sist. Inf.*, vol. 6, no. 1, p. 78, 2022, doi: 10.29103/sisfo.v6i1.7974.
- [5] N. Ramsari and A. Rifaldi, "Design and Build an Academic Activity Scheduling Application Accompanied by a Reminder System Based on Responsive Web Design," *Fig*, vol. IX, no. 1, pp. 1–11, 2018.
- [6] S. N. C. W. Atmaja, R. Oktavianna, S. W. Saputri, P. Purwatiningsih, and B. Benarda, "Time Management for a More Efficient and Effective Life," *STRONG Financial. General and Accountant. Apply.*, vol. 3, no. 1, pp. 60–63, 2021, doi: 10.31092/kuat.v3i1.1165.
- [7] . A. C., . D. A., . I. M., and . N. K., "The Role of Notion Applications in Lectures to Realize Student Productivity," *J. Education, Science and Technology.*, vol. 2, no. 1, pp. 262–273, 2023, doi: 10.47233/jpst.v2i2.754.
- [8] I. D. Niesviantika, H. Marcos, and Riyanto, "Designing Android-Based Schedule and Daily Activity Plan Reminder Applications," *J. Ilm. Tech. Inform. and Sist. Inf. Jl.*, vol. 12, pp. 359–366, 2023, [Online]. Available: <http://ojs.stmik-banjarbaru.ac.id/index.php/jutisi/article/view/1224>
- [9] S. E. Prasetyo, "COMPARATIVE ANALYSIS OF QOS PFSense, OPNSense, AND FLEXIWAN Preliminary Abstraction," vol. 6, no. 2, 2025.
- [10] C. N. Firgiawan, N. Aryasatya, D. F. Ramadan, Y. S. Purwanto, and T. Informatika, "ANDROID-BASED STUDENT ACTIVITY AND ASSIGNMENT MAPPING APPLICATION," vol. 9, no. 2, pp. 2001–2007, 2025.
- [11] B. Ismanto, D. J. Saut HS, and N. Amalia, "Development of an Android-Based Exam Schedule Notification Application," *Rabit J. Teknol. and Sist. Inf. Univrab*, vol. 7, no. 2, pp. 147–155, 2022, doi: 10.36341/rabit.v7i2.2459.
- [12] F. Humam Ramadhan, I. Elan Maulani, N. Faizatuz Zuhriyah, and N. Siti Marlina, "Development of

- Android-Based Mobile Learning Applications to Improve Programming Skills in Informatics Engineering Students," *J. Sos. Teknol.*, vol. 3, no. 2, pp. 108–113, 2023, doi: 10.59188/jurnalsostech.v3i2.638.
- [13] R. Junita Basri and S. Anraeni, "Islamic Information and Technology System Bulletin Designing Lecture Schedule Reminder Application Using a Mobile-Based Push Notification Method," *Bul. Sist. Inf. and Technology. Islam*, vol. 2, no. 1, pp. 52–55, 2021.
- [14] M. Ridhan, M. Nanda, and I. H. Ikasari, "Application of Android-Based Time and Date Merge Activity Schedule Reminder Application," *OKTAL J. Computer Science. and Sci.*, vol. 2, no. 7, pp. 1940–1949, 2023, [Online]. Available: <https://journal.mediapublikasi.id/index.php/oktal>
- [15] A. Maulana, F. B. Wardana, M. I. Hanafri, and N. L. B. Laela, "Gamification-Based Smart Schedule Application to Optimize Time Productivity for Students," *JTIM J. Teknol. Inf. and Multimed.*, vol. 6, no. 2, pp. 168–180, 2024, doi: 10.35746/jtim.v6i2.565.
- [16] I. Larasati, A. N. Yusril, and P. Al Zukri, "Systematic Literature Review of Agile Method Analysis in Mobile Application Development," *System*, vol. 10, no. 2, p. 369, 2021, doi: 10.32520/stmsi.v10i2.1237.
- [17] N. Chandarana and T. Gada, "A Comprehensive Review of Agile Methodologies in Mobile App Development," *Int. Res. J. Mod. Eng. Technol. Sci.*, no. August, 2024, doi: 10.56726/irjmets61006.
- [18] I. Saputra, P. Sukmasetya, and A. Primadewi, "Implementation of Agile Software Development in Waste Management System Design," *CLICK Review. Ilm. Inform. and Compost.*, vol. 4, no. 3, pp. 1930–1942, 2023, doi: 10.30865/klik.v4i3.1379.
- [19] S. Pargaonkar, "A Comprehensive Research Analysis of Software Development Life Cycle (SDLC) Agile & Waterfall Model Advantages, Disadvantages, and Application Suitability in Software Quality Engineering," *Int. J. Sci. Res. Publ.*, vol. 13, no. 8, pp. 120–124, 2023, doi: 10.29322/ijsrp.13.08.2023.p14015.
- [20] N. Abbas, A. M. Gravell, and G. B. Wills, "Historical roots of agile methods: Where did 'Agile thinking' come from?," *Lect. Notes Bus. Inf. Process.*, vol. 9 LNBIP, pp. 94–103, 2008, doi: 10.1007/978-3-540-68255-4\_10.
- [21] M. J. Firdaus and R. M. Manikam, "DESIGN OF TO-DO- LIST APPLICATIONS ' MYLIST,'" vol. 14, no. 164, pp. 36–52, 2025.
- [22] Z. H. Muhamad, D. A. Abdulmonim, and B. Alathari, "An integration of uml use case diagram and activity diagram with Z language for formalization of library management system," *Int. J. Electr. Comput. Eng.*, vol. 9, no. 4, pp. 3069–3076, 2019, doi: 10.11591/ijece.v9i4.pp3069-3076.
- [23] Mr. Huda et al., "Innovative E-Therapy service in higher education: Mobile application design," *Int. J. Interact. Mob. Technol.*, vol. 11, no. 4, pp. 83–94, 2017, doi: 10.3991/ijim.v11i4.6734.
- [24] X. Chen, F. Mallet, X. Liu, F. Verifying, S. Diagrams, and X. Chen, "Critical Systems To cite this version : HAL Id : hal-03121933 Formally Verifying Sequence Diagrams for Safety Critical Systems," 2021.
- [25] S. Sufaidah, "Dan Employee Performance Based on Android Android-Based Employee Performance and," vol. 11, no. 2, pp. 205–214, 2023.