Software Design Document

SleepOnTime

2023-spring-Aberdeen-10

--SIPENG WU (Group Leader)--

--XINYI ZHOU-- --KEXIN LEI--

--HUIQING HUANG--

Content

1 Introduction

1.1 Overall goals

SleepOnTime is a software that provides sleep time monitoring for working people, students and other groups who need to sleep regularly. The product is intended to meet the individual needs of different user groups. By providing an interactive way of APP, we hope that it can restrict the use of specific software during specific hours, calculate sleep time, analyze sleeping condition and other functions, thus helping users to control the time they use their mobile phones at night, improve their sleeping quality, develop in-depth knowledge of their sleep situation and develop a regular routine. The product will be developed using Kotlin as the development syntax and Android Studio as the development tool. Our team will complete the design of all user interfaces and the development of core functions before June 8.

1.2 Overall scope

1.1.1 Project product scope

1.1.1.1 Core functions of application

①Restricting the use of other software for a specific period of time

The user can set a sleep start time and a sleep end time according to their needs, during which SleepOnTime will put the user's phone into a locked state and restrict the use of other software. In exceptional circumstances, the user can use the limited unlocking opportunity to unlock the phone in advanced. Each month the user has three opportunities to unlock the phone. If all three opportunities have been used and there is still a need to unlock the phone, the user will have to watch an advert to get the extra opportunity. This is a simple and effective restriction that significantly reduces the amount of time users spend browsing apps such as Little Red Book, Tiktok and e-novels before going to bed, leading users to develop healthy habits.

②Playing audios to help sleeping

Users can select sleep aids from the library and set the duration of playback to help them fall asleep according to their needs. Sleeping audio will cover a variety of different styles to meet the 4 needs of different user groups, such as pure music, white noise, audio book, etc.

③Lockout time and duration statistic

The software will record the daily start time and duration of the lockout state, analyze and visualise the data on a monthly basis. Users can check their sleep time and adjust their sleep schedule by viewing the records. At the same time, the software will remind users who use the early unlock function several times and encourage them to stick to

their sleep plan.

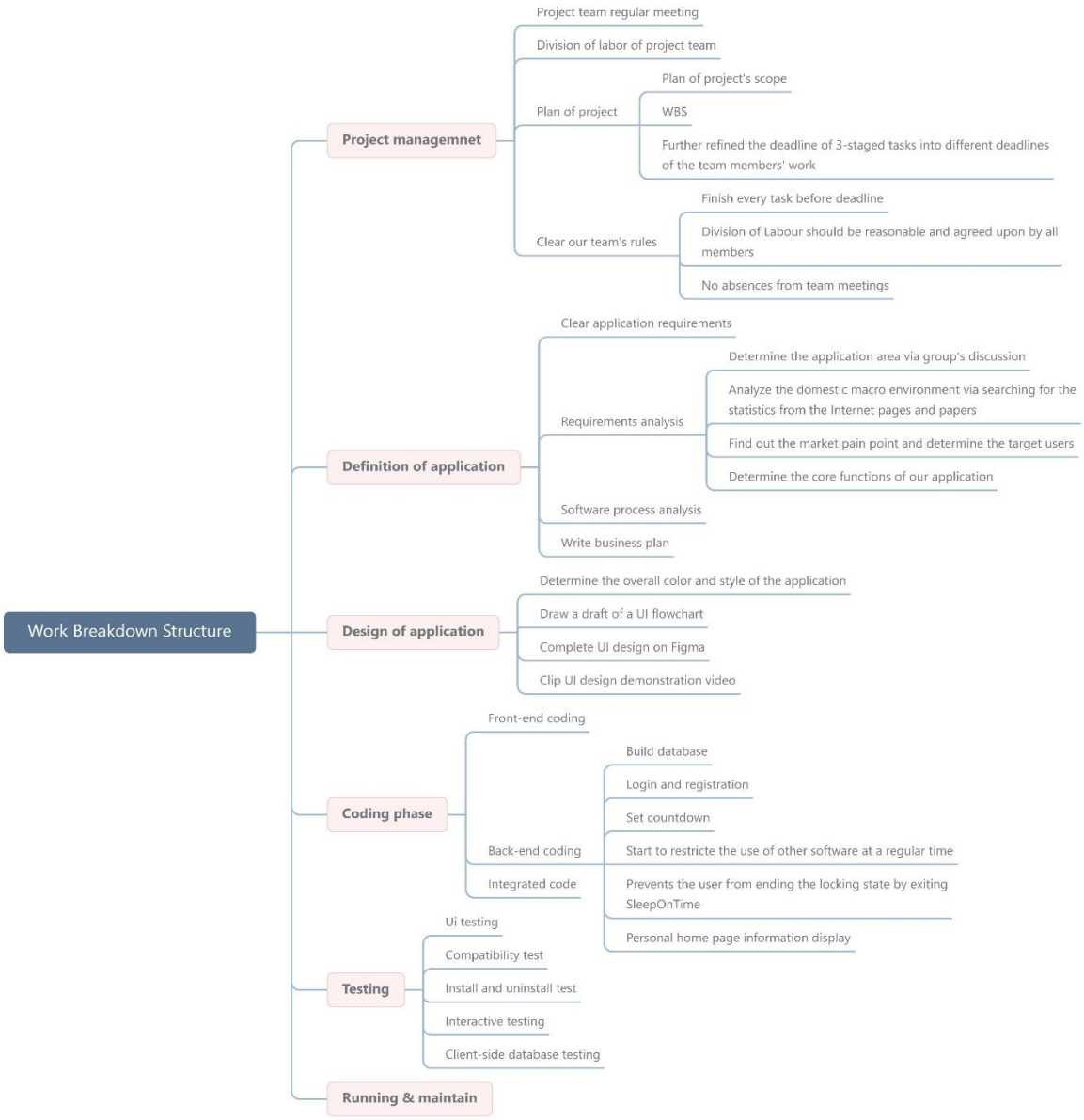
1.1.1.2 Services provided for users

Our application will provide users with installation, registration, login, and users guide to help users get familiar with the software functions.

1.1.1.3 Expected output of our project

Our project plans to develop an Android app, write a business plan and a technical report.

1.1.2 Project work scope



Picture1-1: Detailed work we need to complete

2 Requirements Specification

2.1 User requirements

2.1.1 People's attention to sleep quality and sleep time continues to increase

According to sleep experts and researchers, about half of adults have some degree of sleep problems. At the same time, people are becoming more and more aware of the relationship between poor sleep habits and sleep quality and various health problems, such as psychological problems, obesity, diabetes, cardiovascular diseases, cancer, etc. To raise public awareness of adequate sleep, sleep specialists and doctors actively conduct educational campaigns. For example, many sleep centers, hospitals, and health organizations offer sleep education courses and sleep coaching services that provide services such as sleep assessment and treatment of sleep disorders. In addition, scientific research and advice on sleep and health have been widely reported in various forms of media and social networks. In recent years, although many people still have sleep problems, more and more people have begun to recognize the importance of getting enough sleep and taking positive steps to improve sleep quality.

2.1.2 Mobile phones are a major factor affecting residents' sleep

Using mobile phones in bed before going to bed has become a major factor affecting people's sleep quality. Melatonin is a hormone that regulates sleep, and the blue light emitted by the mobile phone screen easily inhibits the secretion of melatonin in the human body, resulting in the body not receiving sleep signals, difficulty falling asleep or poor sleep quality. Research by Sarah Jane Fox, Ben Carter, and Nicola Lindson shows that the more often you use a smartphone in bed, the shorter the quality and duration of sleep, and the more common it is to have difficulty falling asleep and waking up early.

2.1.3 Contemporary residents have a strong demand for sleep supervision

According to the survey report of the "2018 American Sleep Survey: Electronic Device Use 9 in Bed" released by the National Sleep Foundation in 2018, there are more than 80 Percent said they thought cell phone use negatively affected sleep, and more than 40 percent said they had tried limiting phone use to improve sleep. As a result, many people have taken steps to reduce the disturbance of sleep from phone use, including forcing the phone to turn off before going to bed. However, many people with weak self-control will can't help but turn on their mobile phones again, resulting in a decline in sleep quality, and it is imminent to assist sleep supervision through mobile

phone software.

2.1.4 Existing software can not meet the user’s requirements

At present, the sleep software on the market is defective and cannot solve the user's demands. First, software dedicated to sleep supervision is still in the minority, mostly as a module of a piece of software. When users use software to supervise sleep before going to bed, they are easily attracted by the various functions of the software, so they start browsing and using other functions of the software, and fall asleep later, so that the effect of software sleep supervision cannot be achieved. Secondly, most sleep software in the software market requires users to have a certain degree of self-control, and the effect of sleep supervision cannot be produced for users with poor self-control, and the user's sleep duration cannot be guaranteed. Most of the existing software is unstable during use, and it is easy to be cleared by the background during use, affecting the user experience. In addition, some specific applications of some software require payment. Various deficiencies reduce user satisfaction with sleep monitoring products.

2.1.5 Minimalist interface design

In recent years, most software on the market has pursued a variety of software functions and a full interface design. However, with the intensification of the internal volume, work pressure and academic pressure constantly squeeze the entertainment time of young people today, people gradually tend to use mobile phone software that is easy to use, simple to operate and has a minimalist interface. APPs such as Quark Browser, Curtain and One Word, which mainly focus on "minimalist" design, are rapidly becoming popular. Compared to self-discipline apps such as TomatoTODO and FattyCat, SleepOnTime's interface is designed in a "minimalist" style. The minimalist interface not only improves responsiveness to a certain extent, but also enhances the visual aesthetics by using solid colors to clearly reflect the main functions of the software, helping users to quickly familiarize themselves with the interface and operation of the software. At the same time, the simple operation steps help users to save time on unnecessary entertainment such as decorating personal pages, changing avatars, socialising and so on.

2.2 Functional requirements

2.2.1 Login and registration

The app should give new users adequate guidance when signing up and logging in, while making it as simple as possible for the user to complete the operation. When registering, the user should fill in the verification code to enhance the security of the

user account. You can log in through common social accounts such as Wechat and QQ.

2.2.2 Setting the Lock Time

Setting the lock time is the core function of SleepOnTime. This page should focus on the operability of the application, such as increasing the contrast of background colors and text colors, enlarging the numbers that represent time, and preventing people with poor vision or disabilities from using it normally.

2.2.3 Entering the Lock Mode

After entering the lock mode, a countdown should appear on the page to let the user quickly know how long to end the lock state. Second, when the user wants to end the lock state in advance, the screen should pop up a window to remind the user of the remaining several early unlock opportunities in the month. In order to ensure the function of the lock machine, the correct method of realizing the lock machine should be selected in the process of software development. For example, it is necessary to ensure that the user does not pause the lock countdown when returning to SleepOnTime after opening other software. In UI design, because to achieve the effect of sleep aid, this page must be very warm and comfortable, users will be willing to use this app.

2.2.4 Playing sleep aid audio

First, users should be allowed to choose whether or not to play audio and what audio to play. Secondly, developers should strictly check the audio in this sleep audio library to ensure the quality and effect of the audio. Finally, the playback of sleep audio should be smooth, smooth, and easy to operate.

2.2.5 Going to the Home page

The Home page should be easy to navigate and include statistics on the user's sleep duration and the number of times the device has been unlocked early. Sleep duration data should be visually presented to the user, such as pie charts, histograms, etc. This page should be simple, do not need to have too many complex, fancy features, because these additional, unnecessary features are easy to distract the user, can not achieve the effect of monitoring sleep.

3 Overall design

3.1 Activity

- MainActivity: The entry activity of the app, used for users to choose between registering a new account or logging into an existing account.
- Register: Used for user registration.
- Login: Used for user login.
- Choose: Used for users to choose between initiating sleep lock immediately or setting a timer for sleep lock activation.
- SleepNow: Used for users to select the time when they want to unlock the device after initiating immediate sleep lock.
- SleepLater: Used for users to select the start and end time for sleep lock activation through a timer.
- CountdownReady: Used for users to confirm their selected sleep lock time and initiate the countdown for sleep lock activation.
- CountdownStart: Displays the countdown for sleep lock activation.
- ExoPlayerMusic: Used for users to play soothing sleep music.
- Homepage: Displays the user's personal page, including remaining sleep lock count, total sleep duration, and other information.

3.2 Layout

This application combines Jetpack Compose with traditional XML layout.

- Jetpack Compose is a declarative UI framework for building user interfaces in Android applications. Compared to traditional XML layout, Jetpack Compose offers a new way to create user interfaces using the Kotlin language for UI description and construction.
  - Jetpack Compose is used in SleepNow and SleepLater components in this project.
- Traditional XML layout is also used in this project. It includes ConstraintLayout, LinearLayout, and AbsoluteLayout.
- Since the project primarily uses XML layout, Compose components are embedded within XML layouts, treating them as part of the XML layout.

3.3 Services

- KillProcessService: Monitors recently used applications and closes them when necessary.
- CountdownService: A service responsible for countdown functionality, bringing the application to the foreground and navigating to the corresponding page when a specified time is reached.

3.4 Data Storage

SQLite database is used. A helper class called DatabaseHelper is created to manage the SQLite database, including creating the database, tables, and executing queries and updates.

3.5 Permissions

This application requests the following permissions:

- android.permission.INTERNET: Allows the application to open network sockets for network communication.
- android.permission.KILL_BACKGROUND_PROCESSES: Allows the application to end background processes or applications.
- android.permission.FOREGROUND_SERVICE: Allows the application to run services in the foreground.
- android.permission.SYSTEM_ALERT_WINDOW: Allows the application to display system-level alerts windows, overlaying other applications.
- android.permission.RECEIVE_BOOT_COMPLETED: Allows the application to receive a broadcast after the system finishes booting, enabling the application to automatically start.
- android.permission.REQUEST_IGNORE_BATTERY_OPTIMIZATIONS: Allows the application to request to ignore battery optimizations to keep the application running in the background. android.permission.SCHEDULE_EXACT_ALARM: Allows the application to schedule precise alarms.
- android.permission.REQUEST_USE_EXACT_ALARM: Allows the application to request the use of precise alarms.
- android.permission.PACKAGE_USAGE_STATS: Allows the application to access package usage stats (dangerous permission, requires manual user grant).

3.6 Resources

3.6.1 Components

This application utilizes various components from Material3:

- Button
- Time picker
- Navigation bar
- Text fields
- Dialogs

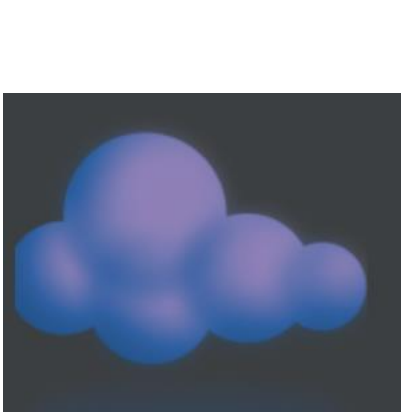
3.6.2 Image Resources

The image resources in this application primarily include:

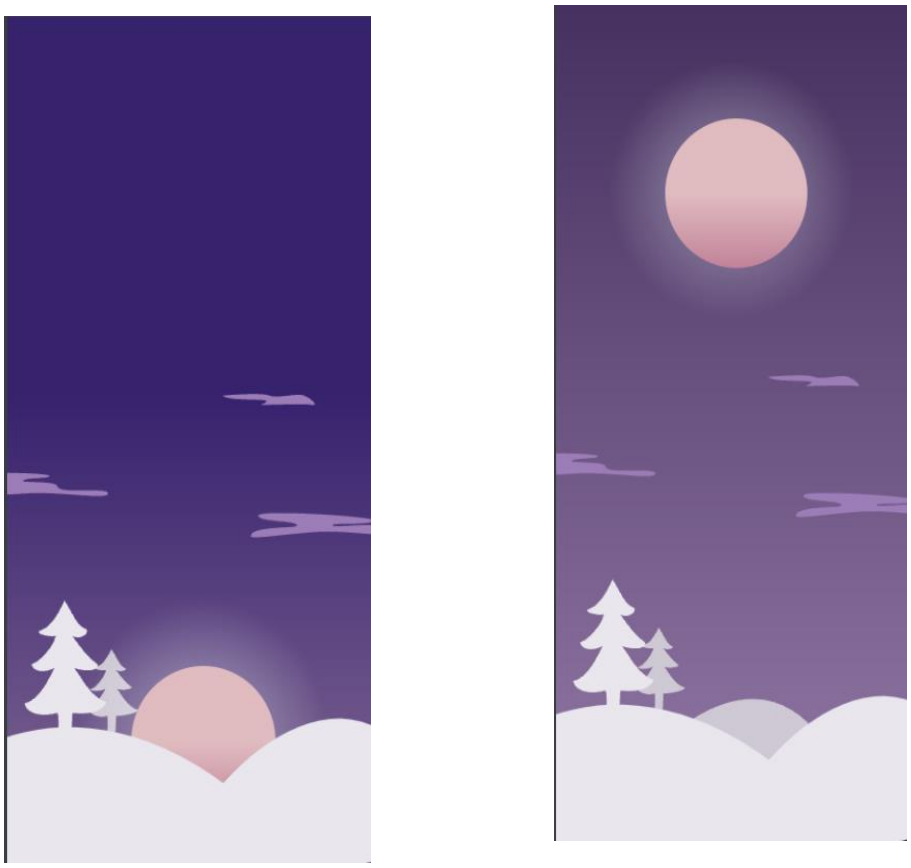
- Background images
- Button background images
- Icon resources

Examples are as follows：

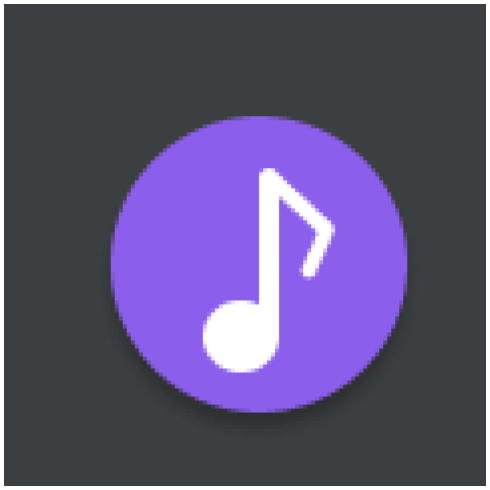
Picture 3-1: Regular page background image



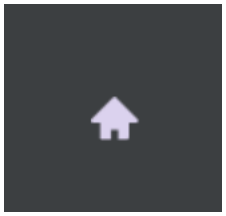
Picture 3-2 Countdown page background decorative element picture



Picture 3-3 Login registration page background image



Picture 3-4 Sleep music button background image



Picture 3-5 Navigation bar Personal home page icon

3.6.3 Color Resources

The main color in this application is purple, and a total of 7 different types of the same shade of purple and white are used. As shown below.

```
<color name="purple_200">#FFBB86FC</color>
<color name="purple_500">#FF6200EE</color>
<color name="purple_700">#FF3700B3</color>
<color name="white">#FFFFFFFF</color>
<color name="add">#ECE6F0</color>
<color name="no_color">#00000000</color>
<color name="purple1">#DBD2EE</color>
<color name="purple2">#D0BCFF</color>
<color name="purple_pink">#D6A9EE</color>
<color name="purple3">#6750A4</color>
<color name="purple4">#513392</color>
<color name="purple5">#37226F</color>
<color name="purple6">#381E72</color>
```

Picture 3-6 Colors used in this application

3.6.4 String Resources

The string resources in this application are primarily used for:

- Text in buttons
- Prompt text in dialogs
- Prompt text in the navigation bar

3.7 Build and Dependencies

3.7.1 Plugins

- com.android.application plugin
- org.jetbrains.kotlin.android plugin

3.7.2 Android Section

- Namespace: The application's namespace is 'com.example.sleepontime'.
- compileSdk and targetSdk: The compilation and target SDK versions are both 33.
- defaultConfig: Contains the default configuration information of the application, such as the application ID, minimum supported SDK version, version code, and version name.
- buildTypes: Defines a build type named 'release' that includes obfuscation settings and ProGuard rules.

- compileOptions: Sets the compatibility of source and target code to Java 1.8.
- kotlinOptions: Specifies the Kotlin JVM target version as 1.8.
- buildFeatures: Enables Compose functionality.
- composeOptions: Specifies the version of the Compose compiler extension as '1.2.0'.
- packagingOptions: Defines packaging options for resources, excluding some files under the META-INF directory.

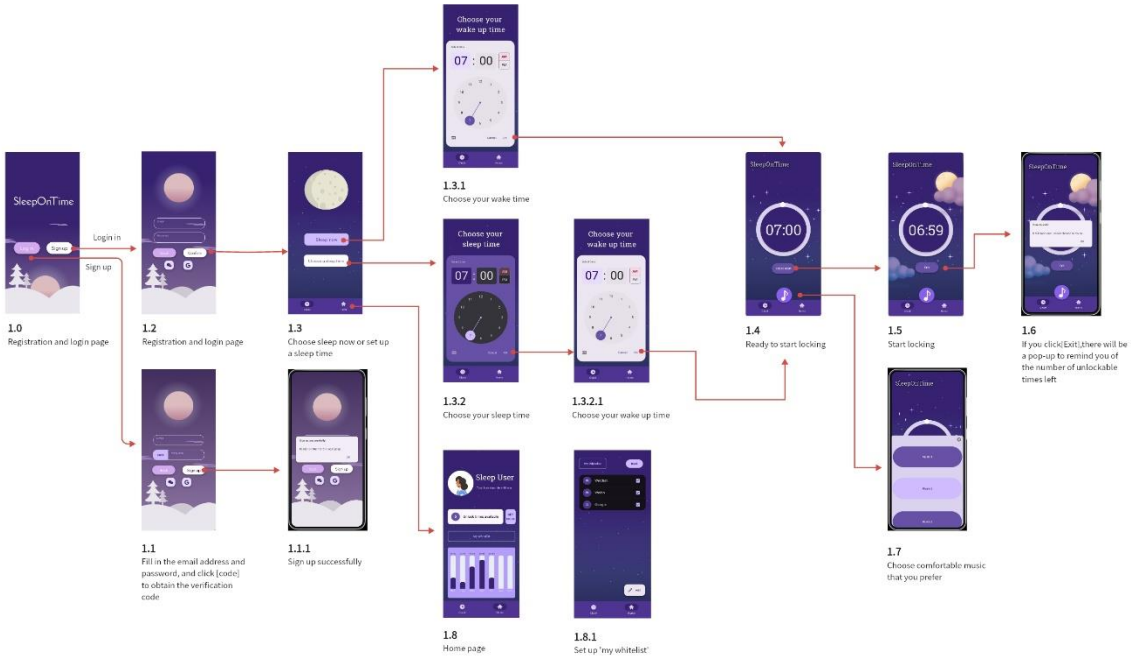
3.7.3 Dependencies Section

- androidx libraries: core-ktx, appcompat, constraintlayout, etc.
- com.google.android.material: Material Design component library.
- com.sun.mail: Android implementation of JavaMail.
- com.android.volley: Library for making network requests.
- org.jetbrains.kotlinx: Kotlin coroutines library.
- com.google.android.gms: Google Play services maps library.
- com.google.android.libraries.maps: Google Maps library.
- junit: Library for unit testing.
- androidx.test: Libraries for Android testing, including JUnit and Espresso.
- androidx.lifecycle: Android lifecycle component library.
- androidx.activity: AndroidX Activity library.
- androidx.compose.ui: Compose UI library.
- androidx.compose.material3: Compose Material3 library.
- com.github.getActivity: Permission request framework.
- com.google.android.exoplayer: ExoPlayer media player library.
- androidx.media3: AndroidX Media3 library.
- org.jetbrains.kotlinx: Kotlinx serialization library.

4 User Interface Design

The UI design is broadly divided into two steps: determining the number of pages, the colour palette, the layout and prototype through the third party platform tools. SleepOnTime's main colour is purple with a minimalist page layout. During the UI design phase, a third-party platform, *Figma*, was used to prototype a total of 14 interfaces. The *material3* component library is also used extensively in prototyping.

4.1 Screen Flow

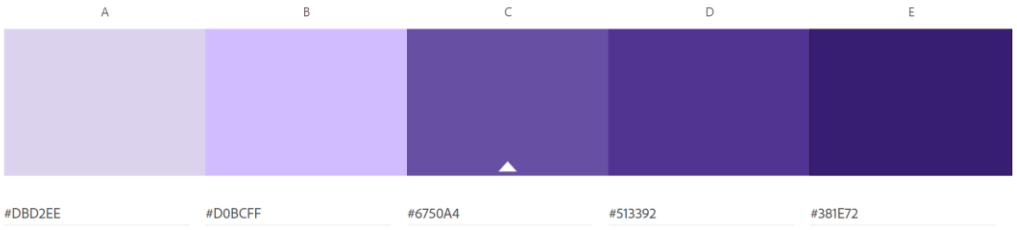


Picture4-1: UI flow

4.2 Color Design

4.2.1 Overview

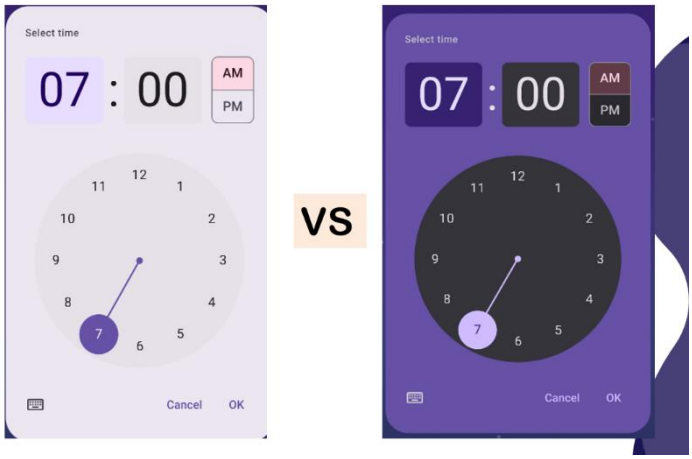
Studies have shown that purple, as a cool colour, has a calming effect on irritability and mind. For stressed and irritable groups such as office workers and students, purple helps to calm the mind and thus helps to sleep. Therefore, SleepOnTime software interface chooses purple as the main colour of the page and presents a harmonious and beautiful interfaces through the combination of various purple colours (Picture 4-2).



Picture 4-2: Main color used in the software

4.2.2 Choose the matching page colour according to the scenario

In the UI design, our team members carefully selected colours to match the scenario of the page. When the user chose the wake up time, our team chose to use a light purple colour, which represents daytime, and when the user chose the sleep time, our team chose to use a dark purple colour which stands for night time (Picture 4-3). The right colour scheme for the page can enhance the user experience to a certain extent and help the user get to sleep as soon as possible.



Picture 4-3: Different colors in different pages according to the scenario

4.2.3 Use colour shades to highlight the main content of the page

Our team members highlight the important content of each page by using the right shades of colour. For example, on light-coloured pages, team members tend to set important information such as text and buttons in dark colours to help users find the important content of the page quickly.

4.3 Layout Design

SleepOnTime uses a minimalist interface design, reducing unnecessary elements in the user interface as much as possible, with minimal buttons and navigation bars to guide users through the software's functions and enable page jumps.

5 Key Technology

5.1 Programming Language

The core logic of the application is written in Kotlin.

5.2 Data Storage

SQLite database is used. A helper class called DatabaseHelper is created to manage the SQLite database, including creating databases, tables, and performing database operations such as queries and updates.

5.2.1 Constants and Static Methods

- DATABASE_NAME: Constant for the database name.
- DATABASE_VERSION: Constant for the database version.
- TABLE1_NAME, COLUMN1_ID, COLUMN1_USERNAME, COLUMN1_PASSWORD, COLUMN1_EMAIL: Constants for the table name and column names of the user table (users).
- TABLE2_NAME, COLUMN2_ID, COLUMN2_USERNAME, COLUMN2_WHITE, COLUMN2_AVAI, COLUMN2_HOUR: Constants for the table name and column names of the sleep info table (sleep_info).

5.2.2 onCreate() Method

Creates the user basic info table (users) and sleep info table (sleep_info) when the database is first created.

5.2.3 .onUpgrade() Method

- Handles database upgrades based on the old and new versions.
- If the old version is 1 and the new version is greater than or equal to 2:
  - Renames the user basic info table (users) and creates a new table.
  - Inserts data from the original table into the new table, then deletes the original table.
  - Creates a temporary table to backup the data of the sleep info table (sleep_info) and inserts it into the new table.
  - Deletes the temporary table and the old table, and creates a new sleep info table.

5.2.4 Database Operation Methods

- insertUserInfo(username: String, email: String): Long: Inserts user information into the user basic info table and sleep info table, and returns the ID of the inserted new row.
- isAvailable(id: Int): Boolean: Queries the remaining unlock count of a user based

on the user ID and returns whether the user is available.

- getUserIdByEmail(email: String): Int: Queries the user ID based on the email and returns the user ID.
- getWeekHour(id: Int): String?: Queries the user's weekly sleep duration based on the user ID and returns it as a string.
- updateWeekHour(id: Int, weekHour: String): Int: Updates the weekly sleep duration of a specified user.
- getUsernameAndAvailableById(id: Int): Pair<String?, Int>: Queries the username and remaining unlock count based on the user ID and returns a Pair object containing the username and unlock count.

5.3 KillProcessService - Main Functionality and Implementation Logic

KillProcessService is a background service used to monitor recently used applications and close them when needed.

①In the onCreate() method, necessary variables and initialization operations are set up:

- Get the package name of the application and save it in the appPackageName variable.
- Create and start a foreground service to ensure that the service runs in the background without being optimized or closed by the system.
- Create a notification channel for displaying notifications in the foreground service.

② In the onCreate() method, Timer and TimerTask are used to perform actions periodically:

- Create a Handler object to handle the logic of the scheduled task.
- Create a Timer object and use TimerTask to execute the scheduled task at a certain time interval.
- The scheduled task sends messages through the Handler to execute the logic.

③Scheduled task:

- Use ActivityManager to get the list of running tasks and retrieve the package name of the top task.
- Use the getRecentAppPackageName() method to get the package name of the most recently used application.
- If the package name of the most recently used application is not the same as the current application's package name or is empty (indicating that the user switched to another application), perform the following actions:
- Start the CountdownStart activity as the entry point of the current application.
- Use the killBackgroundProcesses() method of ActivityManager to close the running application.
- Stop and destroy the service.

④In the onDestroy() method, perform cleanup operations and stop the service:

- Call the stopSelf() method to stop the service.

5.4 CountdownService - Main Functionality and Implementation Logic

CountdownService is a service used to execute countdown functionality. It implements the logic to bring the application to the foreground and navigate to the corresponding page when a specified time is reached.

①In the onCreate() method, create and set up the foreground service notification:

- Call the buildNotification() method to create the foreground service notification.
- Use the startForeground() method to set the service as a foreground service and display the notification.

②In the onStartCommand() method, handle the logic to bring the application to the foreground and navigate to the corresponding page when the time is reached:

- Retrieve the target time from the incoming Intent.
- Create a new Intent and set it as the target to launch the CountdownStart activity.
- Use the startActivity() method to launch the Intent, bringing the application to the foreground and navigating to the corresponding page.
- Other operations, such as starting the countdown, can be performed in this method.

③The onBind() method returns null, indicating that this service does not support binding.

④The buildNotification() method is used to create the foreground service notification:

- Create an Intent for launching the CountdownStart activity.
- Create a PendingIntent and associate it with the CountdownStart activity.
- Use the NotificationCompat.Builder builder to create the notification, setting the title, content, and click action PendingIntent.
- Return the constructed notification object.

5.5 Using the third-party library XXPermissions for permission requests

Since the permissions required by this application, PACKAGE_USAGE_STATS, SYSTEM_ALERT_WINDOW, and SCHEDULE_EXACT_ALARM, are all dangerous permissions that need to be manually granted by the user, the application has chosen the third-party library XXPermissions to simplify the permission request process.

First, in the with(this) block, the .permission() method is called to add the required permissions, PACKAGE_USAGE_STATS, SYSTEM_ALERT_WINDOW, and SCHEDULE_EXACT_ALARM, for the request.

Then, the .request() method is called to initiate the permission request. An anonymous object of OnPermissionCallback is passed as a parameter to handle the result of the permission request.

In the onGranted() method, if all permissions are granted, the allGranted parameter will be true. If only some permissions are granted, the allGranted parameter will be

false.

In the onDenied() method, if the permission is denied and the user has chosen the "do not ask again" option, the doNotAskAgain parameter will be true. If the doNotAskAgain parameter is false, it means the permission was denied but the user did not choose the "do not ask again" option.

5.6 Technical challenges faced during development

① Difficulty sending emails: In the early stages, I was unfamiliar with the PasswordAuthentication function and mistakenly thought that the "password" parameter referred to the email password. As a result, the email sending process was unsuccessful. After researching extensively, I learned that the "password" parameter actually refers to a generated password when enabling SMTP services.

②Difficulty obtaining permissions: This was the most time-consuming challenge during the development of this application. The locking mechanism of this application involved periodically checking the activity stack to verify if the top package name belonged to this application, indicating that it was in the foreground. Consequently, the PACKAGE_USAGE_STATS permission was required.

- This permission is classified as a dangerous permission and cannot be obtained by directly declaring it in the Manifest file. Initially, unaware of its dangerous nature, I declared it in the Manifest file but couldn't access the corresponding functionality.
- In the middle stage, I consulted the official documentation and used the permission request function provided by the system. However, I was unaware at the time that this permission could only be manually granted. Requesting the permission programmatically led to automatic rejection by the system, with the option "Don't ask again" being set. The official documentation did not mention this, so I was stuck in this situation, continuously attempting to find a way to successfully request the permission while relying on the existing code. I didn't realize that using the function to request the permission would lead to automatic rejection, preventing me from obtaining the permission successfully.
- Eventually, by searching online resources, I discovered a manual workaround, which involved directly opening the permission settings page. However, regardless of the method used to manually open the permission settings page (whether through code or by the user accessing the page through the device's settings), the permission check function consistently indicated a failed request. This also consumed a significant amount of time. Ultimately, I'm unsure whether I successfully obtained the permission since, despite manually enabling the permission, the permission check function continued to report a failure.
- Finally, during my research, I came across a third-party permission request framework called XXPermissions, and using this framework, I successfully obtained the required permission.

③Inability to resume countdown upon forced return after exiting:

- The issue here was that upon detecting that the current foreground application was not the intended one, the original code would create a new instance of the application's activity, causing the countdown to restart instead of resuming.

Through extensive research, I learned about the meaning of intent flags and modified the parameters to intent.flags = Intent.FLAG_ACTIVITY_CLEAR_TOP or Intent.FLAG_ACTIVITY_SINGLE_TOP or Intent.FLAG_ACTIVITY_NEW_TASK. This successfully resolved the problem of the countdown restarting upon returning to the application after exiting.

6 Testing and User Experience Analysis

6.1 Test

The test results are as follows.

Test basic information			
Testing application	SleepOnTime	Test pass rate	95.92%
System Platform	Andriod		
Number of models	50		
Number of models not executed	1		

Test result	Number of test terminals	Percent of test result
Installation failure	2	4.08%
Start-up failure	0	0.00%
Monkey failure	0	0.00%
Uninstallation failure	0	0.00%
Run failure	0	0.00%
Pass	47	95.92%

Picture6-1: Test results

6.2 User Experience Analysis

Through user test feedback, basically all users are satisfied with our software. They found our software interface to be simple and clear with an outstanding colour scheme. In terms of functionality, they think our software has all the basic functions, is simple to get used to and is useful for sleep management. At the same time they think our sleep aid music really works to help them sleep. Apart from the advantages, they also made some suggestions about our software. They suggested that they would like to have more sleep music, more third party logins and that the timer would still work when the background is cleared.

7 Conclusion

7.1 Summary

SleepOnTime is a software that provides sleep time monitoring for working people, students and other groups who need to sleep regularly. It can restrict the use of specific software during sleeping hours, helping users to control the time they use their mobile phones at night to improve their sleep quality and develop a regular sleep routine.

We have developed SleepOnTime software, whose basic functions such as timer lock and sleep duration statistics have been largely implemented and have received good feedback from users.

7.2 Challenge

We also encountered a number of difficulties during the development process. In terms of ui design, we were using figama for the first time and were not familiar with this online ui design tool. It took time to get up to speed. Also some components needed to be created by ourselves.

On the front-end development side, we found that the images in figama were not clear when exported and that The colours of the interface designed in figama did not match the colours in Android studio, so it took some time to adjust them. It was also difficult to integrate the final pages as the colours in the interface were too similar.

In terms of back-end development, gaining access to the phone was an insurmountable task that took a lot of time. We also ran into this problem on the technical side. We determined if the foreground application was this application by detecting the application at the top of the activity stack, and if not, closing the foreground application and opening this one. The problem here is that reopening this application restarts an activity rather than opening the original activity, and therefore ends the previous countdown.

7.3 Suggestions

To make our software even better, we will be making these changes in the future Improve the login method and develop more ways to login. This will be enhanced and improved in the future as clearing background processes will turn off the timer. Our music playback function is not stable enough, sometimes it plays music and sometimes it doesn't. We will be doing maintenance and upgrades to fix this.In the meantime we will be developing more useful features to make our software even more functional. In the future, we will also invite original music producers to produce music to enrich our library of songs to attract more users to use our software.