VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**MIDTERM ESSAY**

**MOBILE APP DEVELOPMENT**

**Midterm Essay**

*Instructor*: **LÊ VĂN VANG**

*Executor*: **VŨ MINH QUÂN – 522H0172**

**NGUYỄN ĐỨC HUY – 522H0121**

Class **: 22H50301**

Course  **: 26**

**HO CHI MINH CITY , YEAR 2025**

VIETNAM GENERAL CONFEDERATION OF LABOR

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**

A red and blue logo

Description automatically generated

**MIDTERM ESSAY**

**MOBILE APP DEVELOPMENT**

*Instructor*: **LÊ VĂN VANG**

*Executor*: **VŨ MINH QUÂN**

**NGUYỄN ĐỨC HUY**

Class **: 22H50301**

Course  **: 26**

**HO CHI MINH CITY, YEAR 2025**

**THANK YOU**

Thank you very much, Mr. Le Van Vang, for your dedicated and sincere guidance, support, and assistance throughout the research, exploration, and writing of this report.

.

**COMPLETION OF THESIS AT TON DUC THANG UNIVERSITY**

We hereby confirm that this thesis is our own work and has been conducted under the guidance of Mr. Le Van Vang. The research content and findings presented in this topic are truthful and have not been previously published in any form. The data in the tables, which serve for analysis, comments, and evaluations, have been collected by the author from various sources clearly stated in the reference section.

Futhermore, this thesis includes certain remarks, evaluations, and data from other authors and organizations, all of which are appropriately cited and referenced.

**If any form of academic misconduct is detected, we take full responsibility for the content of my thesis.** Ton Duc Thang University shall not be held liable for any copyright infringement or violations caused by me during the research process (if any).

*Ho Chi Minh City, day month year*

*Author Author*

*(signature and full name) (signature and full name)*

*Nguyen Duc Huy*  *Vu Minh Quan*

**CONFIRMATION AND EVALUATION BY THE SUPERVISOR**

**Supervisor's Confirmation:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, day month year

(signature and full name)

**Evaluation by the Grading Supervisor:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, day month year

(signature and full name)

**SUMMARY**

1. **Introduction problem:**

In the Android ecosystem, App Widgets are an essential feature that displays information & provides functions directly on the device’s home screen or lock screen. Instead of opening an application, users can quickly interact with data directly from the widget, improving the experience and saving time.

1. **Approach**

* **Theoretical research:** Learn how Android supports App Widgets, the structure of a widget, and how it works.
* **Practice creating a simple widget**: Build a basic widget to display static data on the home screen.
* **Performance optimization:** Ensure the widget consumes minimal resources while still functioning efficiently.

1. **Findings**

The basic structure of an App Widget includes:

* **AppWidgetProvider:** Manages the widget’s lifecycle and updates.
* **AppWidgetProviderInfo (XML):** Defines the widget’s size, layout & configuration.
* **RemoteViews:** Provides the interface and handles interactions for the widgets.

Performance optimization:

* Reduce unnecessary updates.
* Limit heavy operations on the UI Thread.

1. **Conclusion**

App Widgets are a powerful feature that enhances the user experience in Android. However, to develop an efficient widget, it is essential to understand its structure, how it works, data update methods & performance optimization techniques.

**TABLE OF CONTENTS**

ACKNOWLEDGEMENTS.............................................................................................1

SUPERVISOR’S CONFIRMATION AND EVALUATION..........................................3

SUMMARY.....................................................................................................................4

TABLE OF CONTENTS.................................................................................................5

LIST OF TABLES, FIGURES, AND GRAPH.................................................................6

CHAPTER 1..........................................................................................................7

CHAPTER 2........................................................................................................11

CHAPTER 3........................................................................................................11

REFRENCE DOCUMENTS..........................................................................................14

**LIST OF TABLES, FIGURES, AND GRAPHS**

**LIST OF TABLES:**

**LIST OF FIGURES:**

IMAGE 1.............................................................................................................11

**CHAPTER I : LEARN ABOUT APP WIDGETS**

1. **What is an App Widgets ?**

App Widget (also known as an Android Home Screen Widget) is a compact user interface component that can be placed on the Home Screen or Lock Screen of an Android device. They allow users to display information or perform quick actions without opening the main application.

Example :

* **Weather widget:** Displays temperature, humidity, and weather forecasts.
* **Note widget:** Allow users to create or view notes without opening the app.
* **Music player widget:** Controls music playback (play, pause, skip) directly from the home screen.
* **To-Do List Widget:** Displays a list of daily tasks.

1. **Why learn & use App Widgets in Android ?**

2.1 Improve user experience

* **Quick access to information:** users can instantly view important information (weather, notes, task lists) without opening an app.
* **Faster interactions:** widgets enable quick actions from the home screen, such as adding a note, playing music, or toggling Wifi.
* **Increased engagement:** apps with widgets are more frequently used since they are always visible on the home screen.

2.2 Increase app visibility & importance

* App with widgets have a higher retention rate as users can interact without opening the app.
* Popular apps like Google Keep, Spotify & Gmail use widgets to enhance user convenience.

2.3 Competitive advantage

* Many apps lack widgets or have poorly designed ones.
* If you provide a well – designed, functional widget, your app will stand out from competitors.

2.4 Better understanding of the Android System

Working with app widgets enhances knowledge of:

* BroadcastReceiver.
* RemoteViews.
* PendingIntent.
* WorkManager / AlarmManager.

2.5 Expand android development skills

* Widgets work differently from Activities or Fragments.
* Learn efficient background processing and data updates without slowing down the system.
* These skills are valuable for IoT applications, automation tools, and home screen utilities.
  1. Business Value & Monetization
* Some apps offer premium widgets as paid features.
* Widgets increase engagement, leading to higher ad revenue or subscription rates.

1. **How does it work ?**

App Widgets operate differently from regular Activities. They do not run continuously but update at specific intervals or based on certain events.

Key components that enable an App Widget to function:

3.1. AppWidgetProvider

* A subclass of BroadcastReceiver that manages the widget’s lifecycle.
* Receives events such as adding, removing, and periodically updating the widget.
* Typically used to update widget data.

3.2. AppWidgetProviderInfo

An XML file (usually located in res/xml/) that defines widget properties, including:

* Default size (minWidth, minHeight).
* Update interval (updatePeriodMillis).
* Display layout (initialLayout).
* Resizing options (resizeMode).

3.3. RemoteViews

Since widgets run outside of an Activity, they cannot use View elements directly. Instead, Android provides RemoteViews to update the widget interface.

RemoteViews allows modifying TextView, ImageView, and handling button clicks via PendingIntent.

3.4. App Widget Host

This is where the widget is displayed on the home screen. Android provides a built-in App Widget Host, but developers can create custom applications that host multiple widgets.

1. **Common Types of App Widgets**

Android supports multiple types of widgets:

4.1. Information Widget

Displays live information (e.g., weather, news, clock).

Example: Google Clock widget.

4.2. Collection Widget

Shows lists or grids of data (e.g., emails, notes, events).

Uses RemoteViewsService for dynamic list updates.

Example: Gmail widget displaying recent emails.

4.3. Control Widget

Allows users to perform quick actions (e.g., toggle Wi-Fi, control music).

Example: Spotify music control widget.

4.4. Hybrid Widget

Combines multiple functionalities (e.g., displaying lists + controls).

Example: Google Calendar widget, which shows events and allows creating new ones.

**5. Benefits of App Widgets**

* **Improved User Experience:** Users can quickly access important information without opening an app.
* **Time-Saving:** Reduces the need to launch apps for simple tasks.
* **Increased Engagement:** Keeps the app visible on the home screen, encouraging more frequent use.
* **Optimized Performance:** Widgets update only when necessary, minimizing resource consumption.

**CHAPTER II : MINDMAP**

**A diagram of a brain lifting weights

AI-generated content may be incorrect.**

**Image 1**

**CHAPTER III : DEMO**

1. **Our Project: Note App  
   A screenshot of a phone

   AI-generated content may be incorrect.**

**A screen shot of a cell phone

AI-generated content may be incorrect.A screen shot of a phone

AI-generated content may be incorrect.**

* My project is about an application where users can record their important and secondary things to avoid forgetting things they need to do.
* About My app: My app includes features such as delete, edit, and add in the add function user can change the font and color of the text, especially a **pin note** feature. The pin button allows users to prioritize important tasks by pinning them. These pinned notes will be displayed on the app's widget, which appears on the home screen. This makes it more convenient for users to check their notes without having to open the app. If users want to edit a note, they simply need to tap on the widget, and it will open the app directly.

1. **How to add a note to widget and how to build a widget**

**2.1 Add Widget Provider to the AndroidManifest.xml**

**A screen shot of a computer program

AI-generated content may be incorrect.**

* 1. **Create widget configuration file (pinned\_notes\_widget\_info.xml)**

**A screen shot of a computer screen

AI-generated content may be incorrect.**

The pinned\_notes\_widget\_info.xml file plays an important role in defining the basic properties and configuration of the widget in your Android application. Specifically, it has the following uses:

- Defines the minimum size of the widget

- Specify widget update frequency

- Specify the initial layout of the widget

- Allows resizing of widgets

- Determine the widget type

* 1. **Create widget layout (pinned\_notes\_widget.xml)**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**A screenshot of a phone

AI-generated content may be incorrect.**

* 1. **Create PinnedNotesWidgetProvider class:**

The PinnedNotesWidgetProvider class inherits from AppWidgetProvider and is the main class for handling events related to your widget. Here are the important methods in AppWidgetProvider and their functions:

There is some important function:

* + onUpdate: Update the interface and handle the event.  
    This method is called when the widget is added to the home screen or when the widget needs to be updated.

A computer screen with text

AI-generated content may be incorrect.

* + onReceive: Handle custom events (if needed). This method is called when any event related to the widget occurs (like update, click, etc.).

A computer screen shot of a program

AI-generated content may be incorrect.

* + updateAppWidget: Updates the widget's appearance and data. This is a static custom method (not part of the AppWidgetProvider class) designed to refresh the widget's UI and synchronize its data. It uses RemoteViews to define the widget's layout and behavior, such as setting up a clickable area to launch MainActivity and binding a RemoteAdapter to display pinned notes in a ListView. It takes Context, AppWidgetManager, and appWidgetId as parameters to target and update a specific widget instance. In this application, it ensures the widget reflects the latest pinned notes and supports user interaction, working alongside broadcast intents and AppWidgetProvider lifecycle methods like onUpdate.A screenshot of a computer program

    AI-generated content may be incorrect.
  1. **Create RemoteViewsService to provide data:**

Create a PinnedNotesWidgetService.java file to provide a list of pinned notes:

* The PinnedNotesWidgetService and PinnedNotesWidgetFactory classes play a crucial role in providing data (a list of pinned notes) to the widget. Below is an explanation of the key methods in RemoteViewsService and how they are applied in this application:
  + **PinnedNotesWidgetService**: This class extends RemoteViewsService and is responsible for creating a RemoteViewsFactory to provide data to the widget.

A screen shot of a computer code

AI-generated content may be incorrect.

* + - Purpose: This method is called when the system needs a RemoteViewsFactory to provide data for the widget.
    - Application in the app: In this app, it returns an instance of PinnedNotesWidgetFactory to provide the list of pinned notes.
  + **PinnedNotesWidgetFactory:** This class implements RemoteViewsService. RemoteViewsFactory is responsible for providing data to the widget.

**A computer screen shot of a program

AI-generated content may be incorrect.**

* + - Purpose: This method is called when the data needs to be updated (e.g., when the widget requests a refresh).
    - Application in the app:
      * Reads the list of notes from SharedPreferences.
      * **SharedPreferences:** Is a lightweight storage mechanism in Android that allows you to save and retrieve small amounts of data in the form of key-value pairs. It is commonly used to store user preferences, settings, or simple data that needs to persist across app sessions.
      * In the **Sticky Note** app, it is used to save and retrieve the list of notes as a JSON string, making it easy to persist data across app sessions.
      * Uses Gson to convert JSON data into a list of Note objects.
      * Filters out pinned notes (isPinned() == true) and adds them to the pinnedNotes list.
    - A screen shot of a computer

      AI-generated content may be incorrect.
      * Purpose: This method is called when the RemoteViewsFactory is destroyed.
      * Application in the app: Clears the pinnedNotes list to free up memory.
    - A screenshot of a computer screen

      AI-generated content may be incorrect.
      * Purpose: Returns the number of items in the data list.
      * Application in the app: Returns the number of pinned notes in the pinnedNotes list.
    - A computer screen shot of a program code

      AI-generated content may be incorrect.
      * Purpose: Returns a RemoteViews for the item at the specified position in the list.
      * Application in the app:
        + Creates a RemoteViews from the widget\_note\_item.xml layout.
        + Sets the title (title) and content (notes) of the note into the corresponding TextViews.
        + Applies the note's color (color) to the title and content.
    - A screen shot of a computer code

      AI-generated content may be incorrect.
      * Purpose: Returns the number of different view types in the list and the ID of the item at the specified position.
      * Application in the app: Returns 1 because there is only one type of view (the widget\_note\_item.xml layout).
      * Returns position as the ID.
    - A black background with blue and orange text

      AI-generated content may be incorrect.
      * Purpose: Returns true if the IDs of the items are stable (do not change when data changes).
      * Application in the app: Returns true because the IDs of the notes do not change.
  1. **Update MainActivity to notify widget when list changes**

A computer screen shot of a program

AI-generated content may be incorrect.

* **Purpose of the updateWidget Function:**
  + **Update the widget's data**: When the user performs actions such as adding, deleting, or pinning/unpinning a note, the data in the notes list changes. This function ensures that the widget displays the latest data.
  + **Send an update signal to the widget**: The function sends a broadcast with the action AppWidgetManager.ACTION\_APPWIDGET\_UPDATE to notify the widget that the data has changed and needs to be refreshed.
  + **How the updateWidget Function Works:**
    - **Create an Intent:** Create an Intent with the action AppWidgetManager.ACTION\_APPWIDGET\_UPDATE
      * This action notifies the widget that it needs to be updated.
    - **Get the List of Widget IDs**: Retrieve the list of IDs for all widgets currently displayed on the home screen
      * AppWidgetManager manages the widgets and provides the list of IDs for widgets belonging to the PinnedNotesWidgetProvider class.
    - **Send a Broadcast**: Send a broadcast with the list of widget IDs that need to be updated
      * EXTRA\_APPWIDGET\_IDS: Contains the list of widget IDs that need to be updated.
      * sendBroadcast(intent): Sends the update signal to all widgets with IDs in the list.

* **Summary:**
  + The updateWidget function:
    - Notifies the widget that the data has changed.
    - Ensures the widget displays the latest data from the notes list.
    - Is called whenever there is a change in the list of notes (add, delete, pin/unpin).

**REFERENCE DOCUMENTS**

* **Chapter 1:**

<https://www.thegioididong.com/game-app/widget-la-gi-tim-hieu-ve-widget-tren-dien-thoai-android-va-1362662>

<https://viblo.asia/p/xay-dung-android-home-screen-widgets-voi-appwidgetprovider-RQqKLeepZ7z>

* **Chapter 2:**