**CONTENT**

**Table content 1**

**Preface 2**

**Introduction 3**

1. Main reason for choosing the topic 3

2. Purpose of the research 3

3. The research’s scope 3

4. Research methodologies 3

5. The main contents of project 3

**Chapter 1: Android and Android Studio overview 4**

1.1. Android 4

1.2. Android Studio 4

**Chapter 2: Overview of Sliding Puzzle Game projects 5**

2.1. Introduction and rules about sliding puzzle game 5

2.2. Project build 5

**Chap 3: Difficulties and advantages during the project implementation 16**

3.1. Difficulties during the project implementation 16

3.2. Advantages during the project implementation 17

**PREFACE**

After a period of studying and researching with observations and summation at the Ho Chi Minh City University of Technology. Especially, the enthusiastic help of a teacher (teacher) Huynh Dang Chieu Phu helped us to complete the report on Java Programming.

We would like to thank teacher Huynh Dang Chieu Phu, the Ho Chi Minh City University of Technology for providing background knowledge of the subject, who directly guided us to complete the report on Java Programming.

**INTRODUCTION**

1. Main reason for choosing the topic

During our studies, most of us just learned to make apps, but now the school has created conditions for us to make games on Android mobile devices, although apps and games are very different, because we want to challenge ourselves, gaining more experience, as well as realizing the dream of making a game is a simple game, it's sliding puzzle game, but it helps us learn many things in the process of making that game and teamwork skills.

1. Purpose of the research

The purpose is: improve our own ability and apply what we have learned and practiced, gain more experience and other skills such as soft skills, teamwork skills, google search, logical thinking, Ability to write code… These will be the best preparation and support for our future path.

1. The research’s scope

Our scope of research is the basics of the game expressed in the Android mobile application language that is part of the Java programming language and the Android Studio framework. How a game works, the structure of its classes and structures, how the game loop works, and how the game handles and responds to the needs of the players.

1. Research methodologies

Our research method is one person at a time. The team has 3 members, the team leader will oversee deciding how many parts the game will have and related things, is the main coder for the game. The remaining 2 members will oversee finding information and documents as complete as possible to support the team leader, in which 1 person will find and filter the information that the team leader needs to write code, the other will research, carefully study the syntax, commands, functions of that command and parameters and record the difficulties in the implementation process to write a report later. With this method, the work will be divided equally among all 3 people and each person will receive an almost equal amount of knowledge or skills.

1. The main contents of project:

* Chapter 1: Andrdoid and Android Studio overview
* Chapter 2: Overview of Sliding Puzzle Game projects
* Chapter 3: Difficulties and advantages during the project implementation

**Chapter 1: Android and Android Studio overview**

Android is the main programming language and Android Studio is the main support tool in the project implementation process.

* 1. Android
* Java is the official language for Android development. This is the language most supported by Google. This is also the language in which most Play Store apps are built. It is also the official language of Android. Google's Android operating system uses Java as the foundation for all Android applications.
* Java is designed to be compatible with many development environments, so it is considered to be more flexible than the object-oriented programming language C++ and the C programming language. In addition, Java has high-performance thanks to its high-performance collection set garbage. It frees memory with unused objects. Because if we already know the Java language, this is advantage for us when choosing to learn Android programming.
  1. Android Studio
* Android Studio is the official environment for the development system (IDE) for Android development apps. Where developers code and assemble their applications from packages and other libraries.
* Library software, utility setup tools more to help us build and test. As well as debugging Android apps.
* Android Studio supports a wide range of emulators for app previewing, so even if we do not have device testing, we can be sure that everything is working smoothly.
* In addition, a series of companies such as optimization advice sales graphs, and data derived from the analysis will help app developers manage their sales and find the tools that come with each Android device.
* Android Studio supports Windows, Mac OS X, and Linux operating systems, and is the official Google IDE for developing native Android apps as an alternative to Eclipse-based Android Development Tools (ADTs).

**Chapter 2: Overview of Sliding Puzzle Game projects**

* 1. Introduction to sliding Puzzle Game and the game rules:
     1. *Introduction and rules*
* Sliding Block Puzzles are made up of a set of shapes that are placed inside a frame. Within that frame, the shapes can be moved around only by sliding, with NO turning, lifting, or jumping allowed. Usually, the starting positions of the shapes are given (but might be chosen at random) together with the finishing positions which must be achieved to solve the puzzle.
* Though it was not the first of this type of puzzle, the one that really started it all off was the 15 Puzzle (described below) which was invented (by an unknown person) in 1878 in the USA and which, very quickly, spread to many other countries. It was marketed under various names - 'The Boss Puzzle' was one of them. The craze for this puzzle lasted about 3 years and several millions of people took it up. The supply could hardly keep up with the demand, and it was reported that one shop alone in New York was selling over 33,000 copies a day. All of this sounds very much like the craze for Rubik's Cube which took place almost exactly 100 years later.
  + 1. *Play mode*
* This puzzle shown on the right has n numbered square blocks that are free to slide inside a n x n frame depending on the player's choice. For example, if we choose 4x4, there will be 16 squares, but 1 square will be lost.
* To start, all the blocks are taken out and then replaced at random.
* The object then is to re-arrange them into their correctly ordered sequence (as shown) simply by sliding them around within the frame.
  + 1. *Some example images sliding puzzle game*

A picture containing electronics, calculator, keyboard

Description automatically generated

* 1. Project build

Based on a predefined pure 4x4 game mode, we have built more interesting points to the game. For example, adding 3x3, 5x5 mode, adding picture.... and time limit in each different game mode, to increase the challenge and decisiveness of the player.

* + 1. *What is the idea before implementation?*

At first, we intended to make a great game, but after discussing the ability of each team member, we did not have enough knowledge to be able to deploy. In the end we chose simple games like Tetrix, puzzle, slappy bird.... and agreed to choose Sliding Puzzle Game. The original structure of the Slide Puzzle Game includes 5 states for Main menu:

- State Flash Screen will have:

+ Button fine-tune music and volume.

+ New Game

+ Score Board

+ Exit

After clicking on State Flash Screen, the levels will appear at:

- State Mode Game (Choose the mode play):

+ 3x3

+4x4

…

After selecting the mode, you will enter:

- State In Game:

Request again the user mode selected in State Mode Game and the corresponding time interval for each selected mode.

- State Win Game:

After the user solves the puzzle, the State Win Game will appear at:

The You Win line will appear and:

+ Score: replays the time that the user has left after the Win Game.

+ Play again.

+ Back to the menu.

+ Exit.

- State Lose Game:

After the allotted time has elapsed, but the user has not solved the puzzle, the You Lose line will appear and:

+ Play again.

+ Back to the menu.

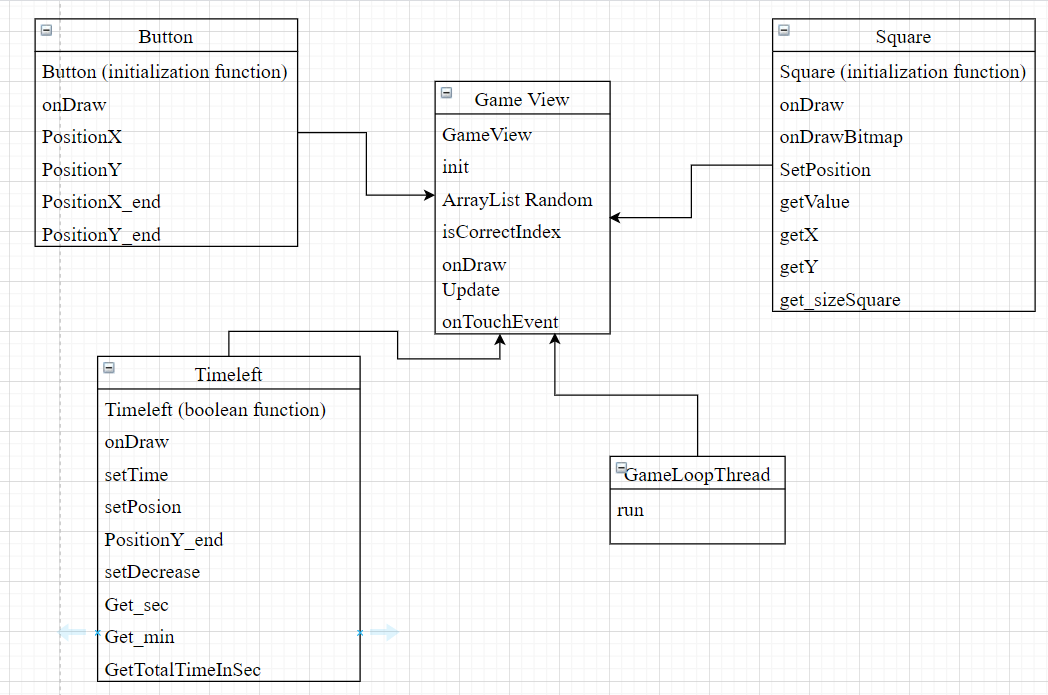
+ Exit.

* + 1. *Project implementation process*

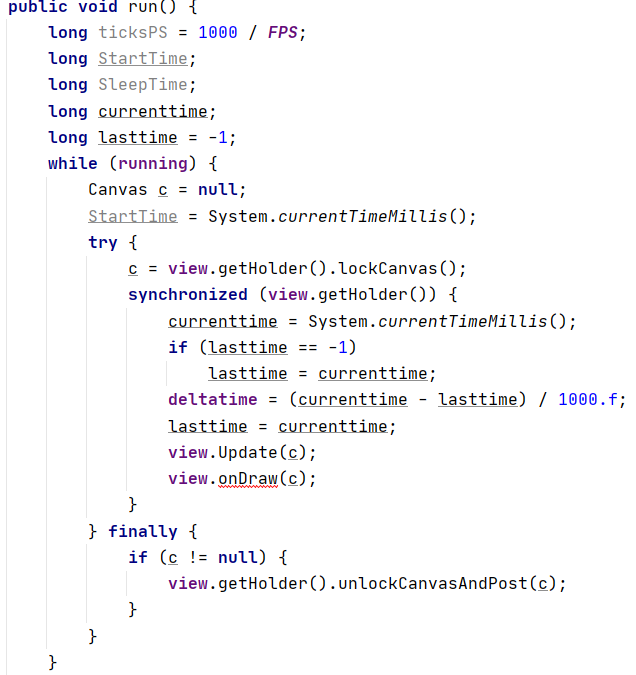
Building on these pre-existing ideas, we started to build the game with the following steps:

Step 1: We proceed to build classes with function and method are the most important methods in the game, without them there would be nothing. We have all 5 classes built into the game:

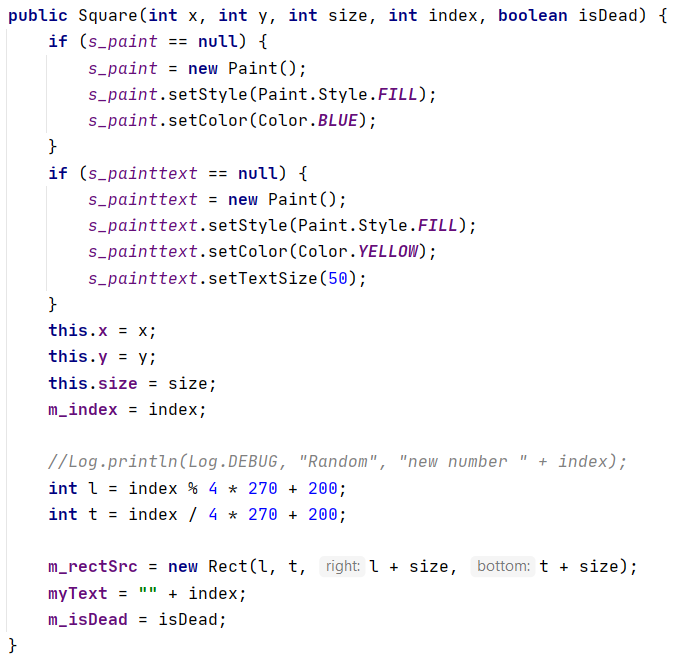
* Class Game Loop Thread is considered the "pile-line" component of the game, game changes will be updated and redrawn in each iteration.
* Class Square is the class that builds the squares in the game and includes constructors and methods that take values in the class that do the processing inside the GameView.
* Class Button is the class that builds the buttons in the game includes constructors and methods that take values in the class that do the processing inside the GameView.
* Class Timeleft is the class that builds the buttons in the game includes constructors and methods that take values in the class that do the processing inside the GameView.
* The GameView class is the "heart" component of the game where the user's buttons and interactions are handled on the squares that have been initialized by the class through the game, as well as the game timing.



Here are some related images of the classes available in the game (Class Square and Class GameLoopThread in the project):

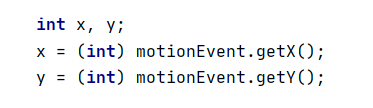


(This is an endless loop that happens throughout the game, until the game is closed, and is the biggest difference between applying for a job and playing the game).

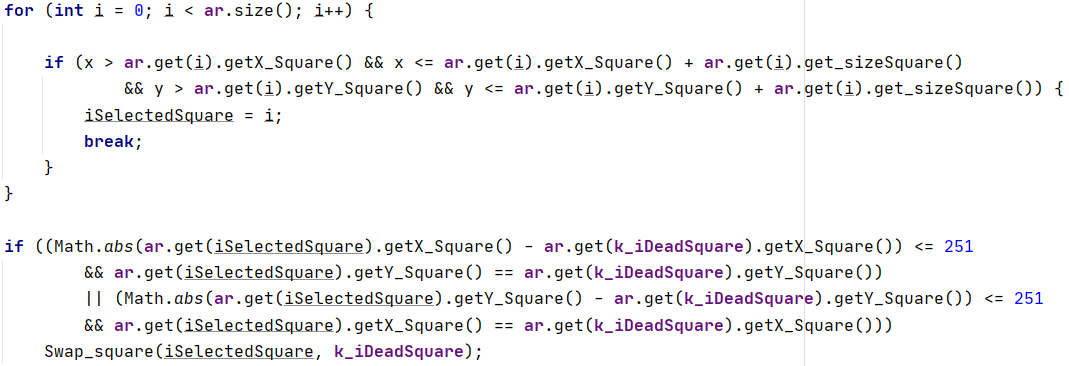


(Here is the constructor that decides how the square will be made)

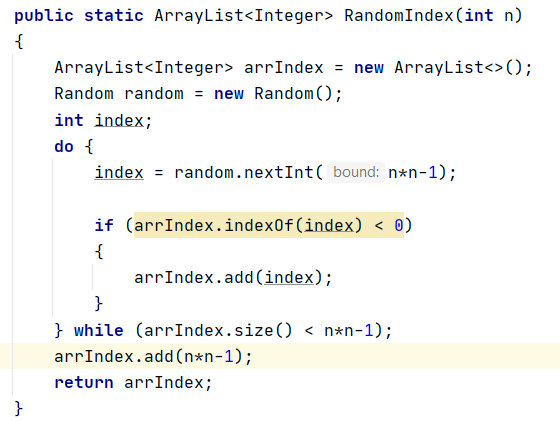
* Step 2: Handle buttons and user interactions through the game (We will handle user interactions through Class GameView)
* The interaction between the user and the game is extremely important, to be able to get the coordinates that the user has interacted with on the game, it is necessary to determine the position of interaction by:

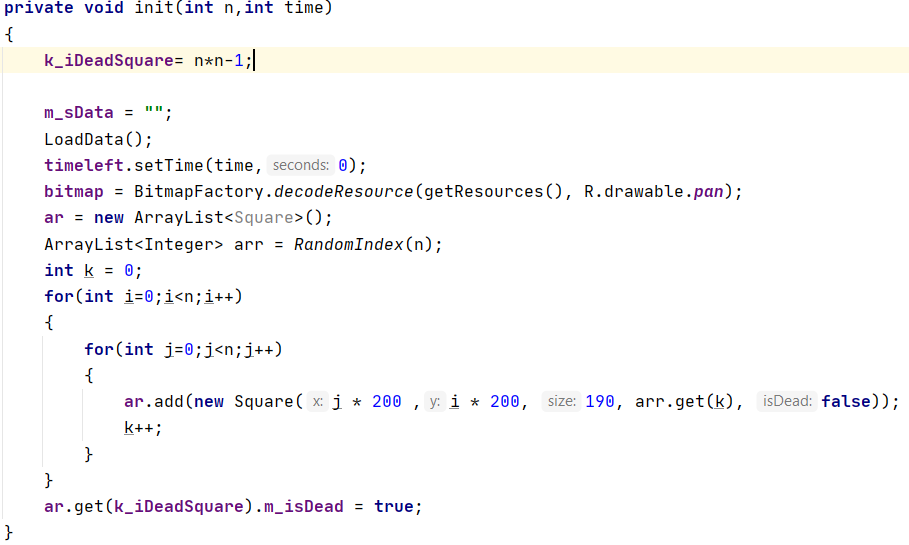


* After you have done that, handling things like moving the square up and down left and right will be very simple. However, it still depends on the rules of the game when the square can only move through places where no squares are in the way.

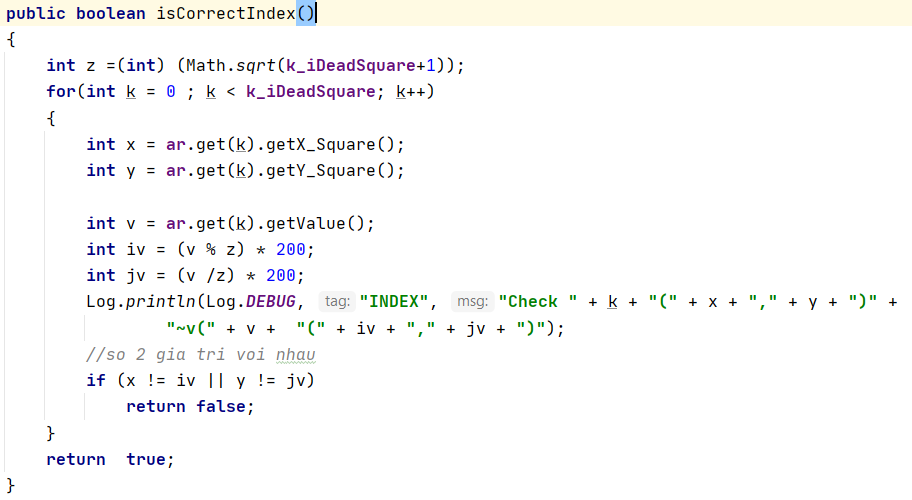


* But before moving the squares, there must be an initialization step as well as shuffling the position of the squares to form a puzzle. Each player-selected matrix will have a dead position (k\_iDeadSquare).

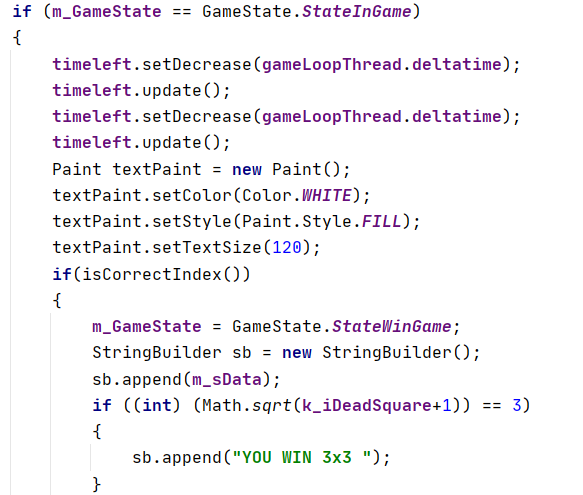




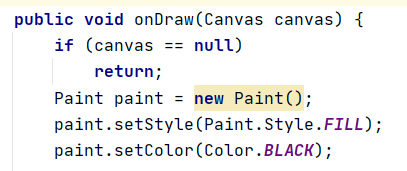
* To be able to win the game, we must check if the players have arranged the correct position or not. Here will consider 2 positions: 1 is the actual position of the square, 2 is the position of the square that has been turned upside down. If indeed they were in the right place where they belonged before the allotted time, the player has won.



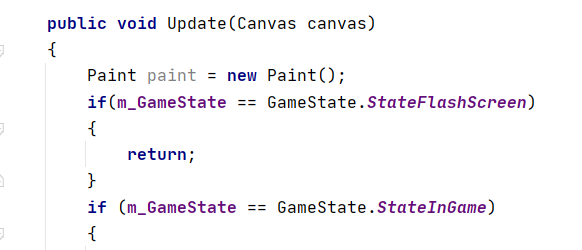
* Time left is an interesting feature, it can decide the person who will win if he solves the puzzle before the countdown time end and will lose if he does not solve the puzzle before the countdown time end.



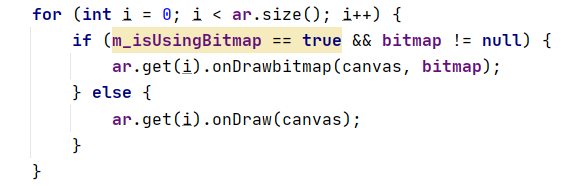
* To be able to draw a square, the onDraw function is indispensable.

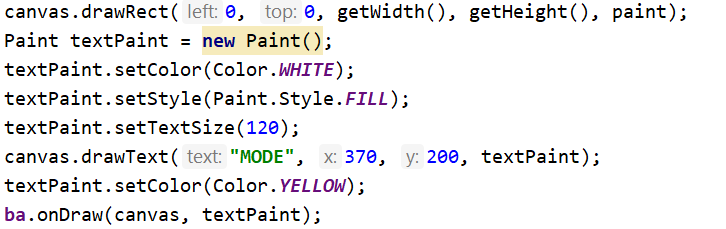


* To be able to draw a square, the Update function is indispensable.

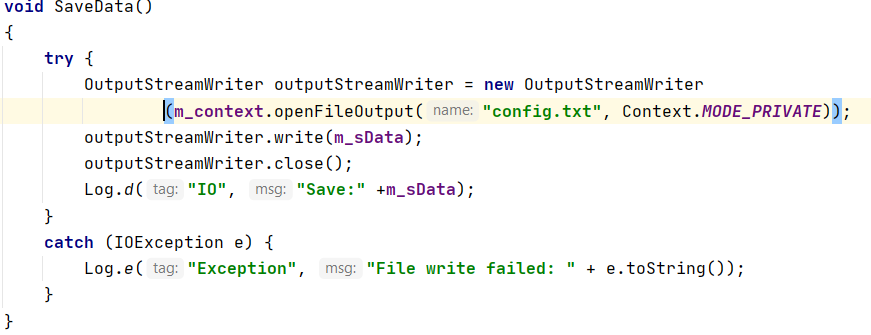


* onDraw() and update() are two good functions, it helps to keep the code well organized, separate drawing, separate logic -> easy to debug/maintenance and make the code in onDraw/update of GameView neat and easy to read.
* Being in the onDraw function can help us draw squares as well as color them and buttons.





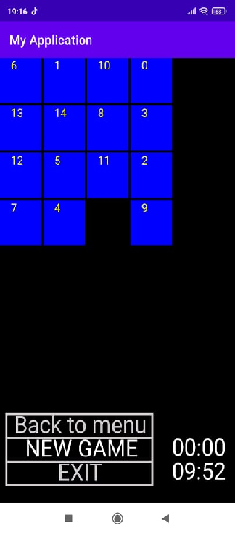
* There is also a save and load feature to help users save their winning moments in the game on the leader board.





* Step 3: Double check all existing functions in the game and complete the project again before ending the project.

Here are some pictures after the project is completed:

**Chap 3: Difficulties and advantages during the project implementation**

1. Difficulties during the project implementation

* The first difficulty is coming up with an initial idea for the project, we still haven't figured out how to make a game using algorithms. Although the processing structure between making applications and games is similar, the game maker will focus a lot on the object than applications.
* In construction, there are problems caused by algorithms, functions, and variables with different calling. Although all programming languages have a similar structure, each language has its own way of expressing it.
* The biggest inadequacy in game making is that we still haven't figured out what the rules of game making are? What is gameloop? Why use gameloop? How does the gameloop algorithm work?
* Each member acquires the same knowledge, but each person will have a different way of thinking, leading to disagreements during project implementation.
* Thinking about algorithms: there are things that we have learned but have not used for a long time, making us forget, causing a lot of difficulties in structuring functions, methods, etc.
* In addition, there are objective problems such as: power outage, illness, incorrect information search, .... leading to delay in the project progress.
* Unnecessary errors appear and do not know where to check that error comes from to correct it in time, leading to the delay of the project.

1. Advantages during the project implementation

* *Technology*
* Nowadays, finding information becomes easier with support tools and websites such as google.com, stack overflow, geeksforgeeks.org, etc.
* The Android studio framework has also greatly supported the process of coding, bringing images and sounds into the project, help find bugs in project. As well as the java language, especially android, it is not too complicated for beginners to learn to make games.
* *Members of the group*
* Although each person will have a different way of thinking, but everyone is for a common purpose, supporting each other, overcoming each other's shortcomings in the process of project implementation.
* Team members also have backup plans in case of force majeure. For example, if a member's house has a power outage and cannot continue with the project, other members will take turns studying what that member is doing to respond promptly and complete the project on schedule.