ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KỸ THUẬT MÁY TÍNH

BÁO CÁO CUỐI KỲ

MÔN THIẾT KẾ HỆ THỐNG NHÚNG – CE224

HỌ VÀ TÊN	**Vòng Chí Cường** **Vũ Hoàng Tuấn** **Trần Hoàng Gia Huy**	**21521910** **21521640** **21520936**
LỚP	**CE213.O11**	

GIẢNG VIÊN HƯỚNG DẪN

Trần Ngọc Đức

TP. HỒ CHÍ MINH – Tháng 1 năm 2024

Table of Contents

List of Figures

I. Introduction

In the current era of technological advancement, service and management systems have become more modernized. A notable example is a parking management system utilizing RFID technology and web services. This project aims to implement a vehicle parking management system that incorporates RFID technology for vehicle identification and a web-based service for efficient administration.

II. Technology Stack

Hardware	STM32F411CEU6 ESP32-CAM Ai-Thinker
Services	Firebase Cloud Firestore Authentication
Framework	Flutter
Programming language	Dart C / C++

Table 1 - Technology Stack

III. Architecture Design

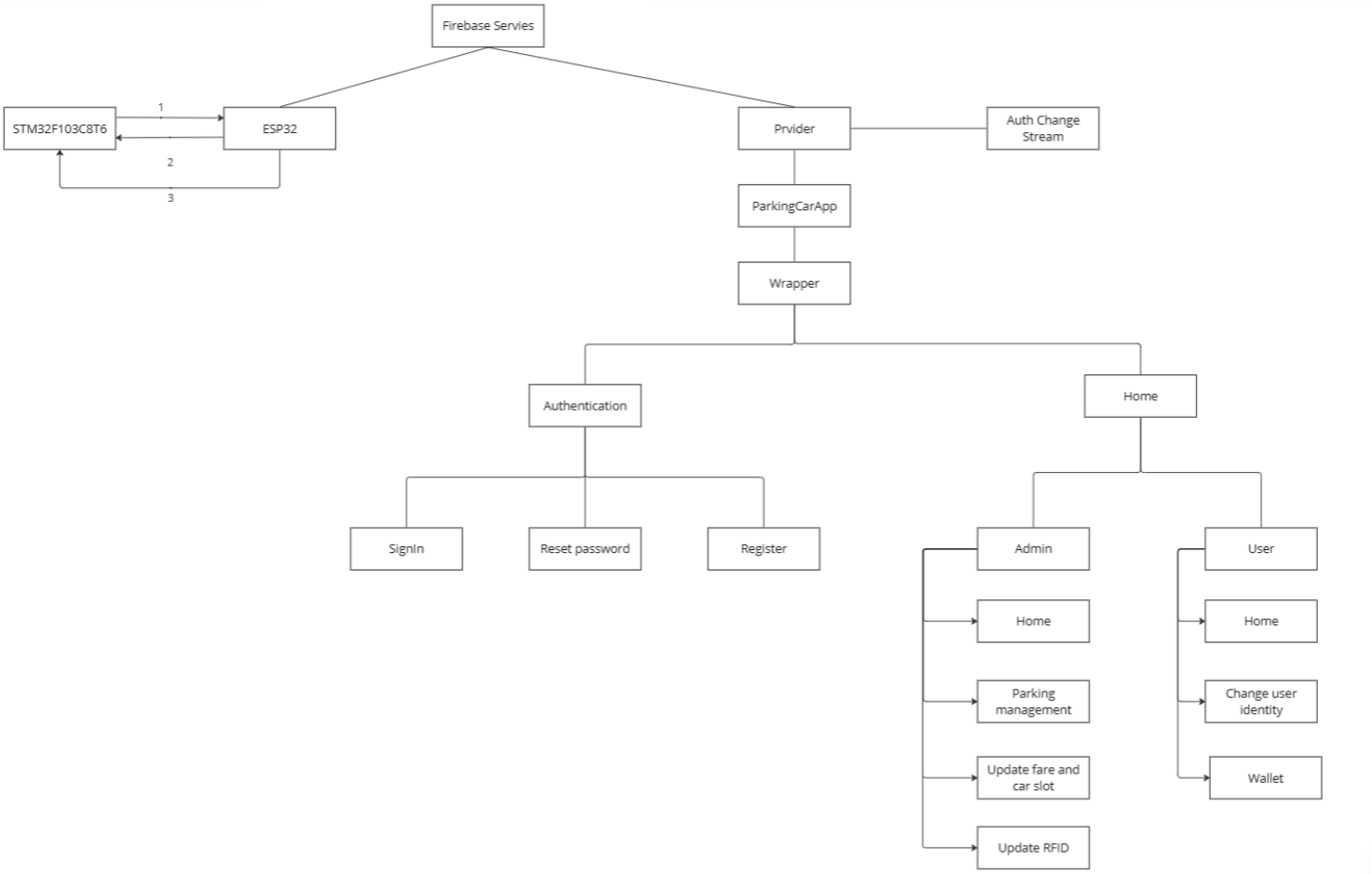


Figure 1 - Architecure Design

Description: *Diagram representing the connection architecture between hardware, database and software of the parking app*

Keynote

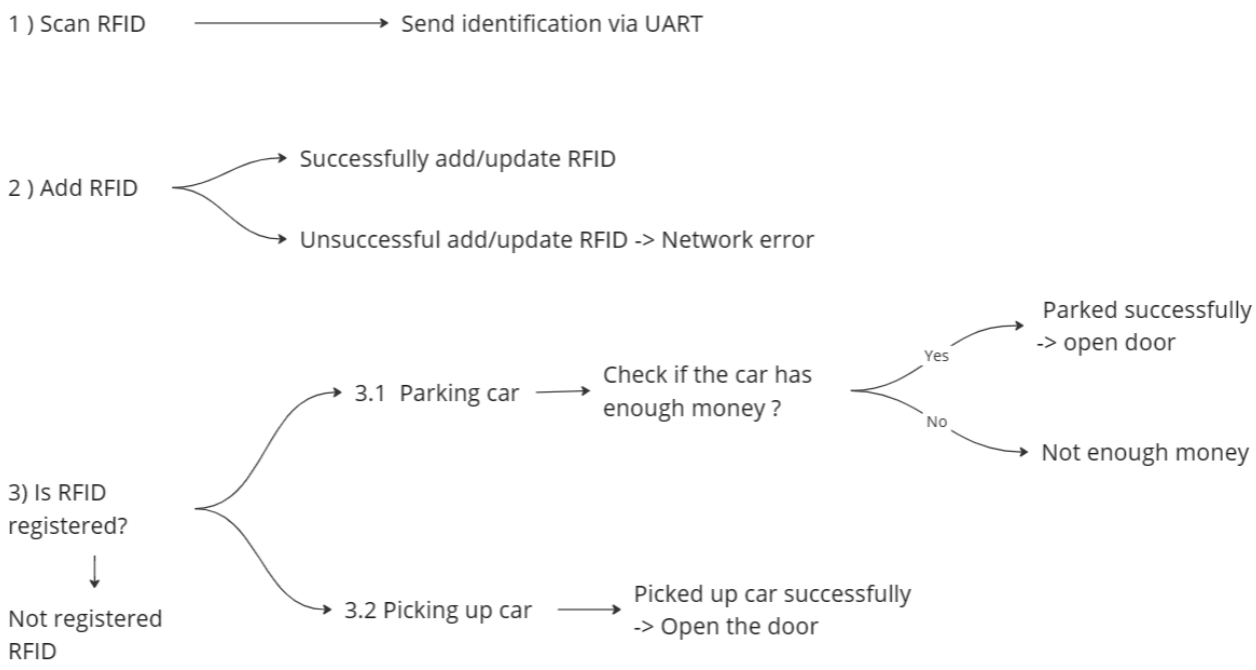


Figure 2 - Keynote

Description: *This image is used to explain states 1,2,3 in the above architecture diagram*

State Diagram

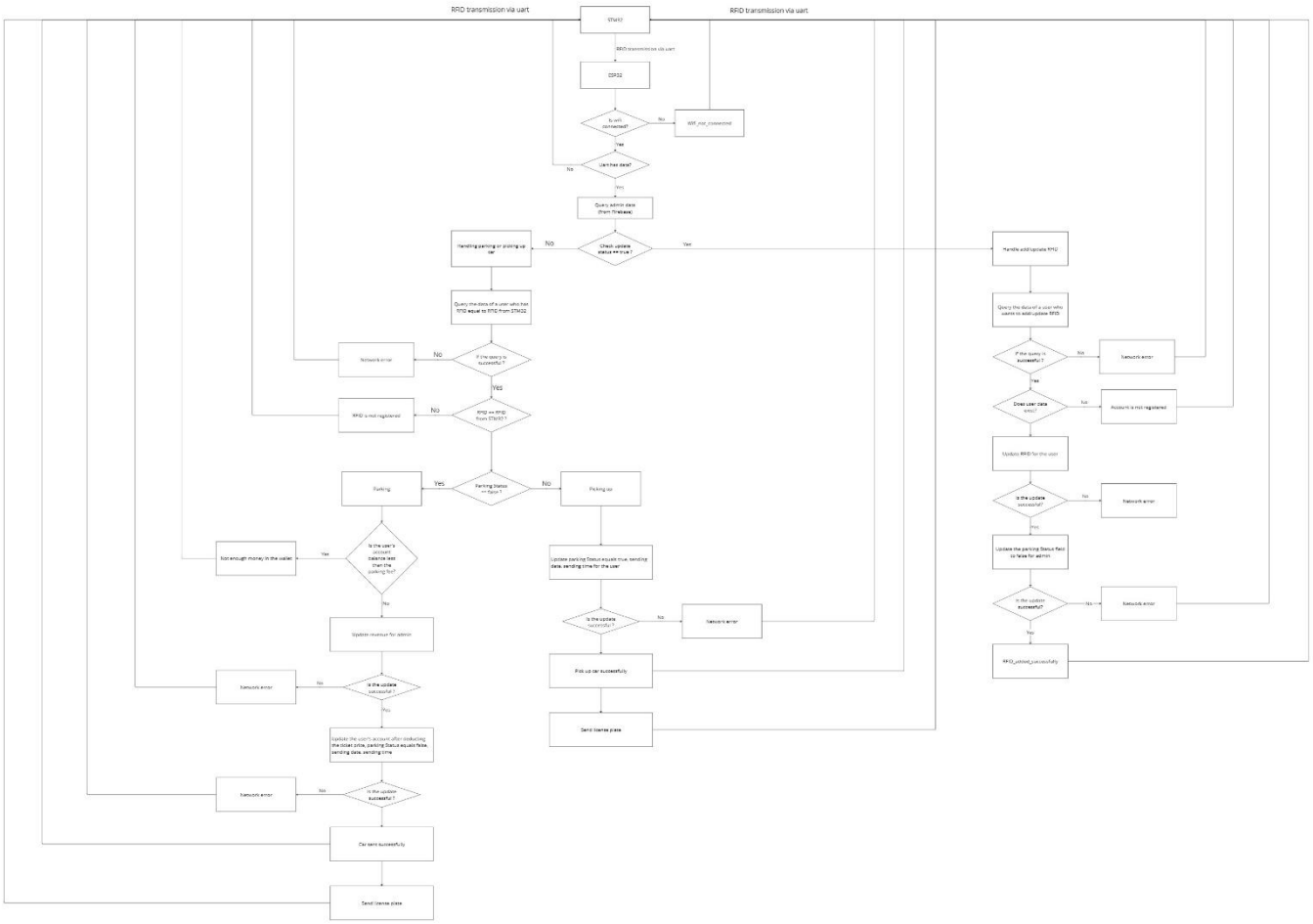


Figure 3 - State diagram

Description: *Diagram showing the status of signal processing when receiving RFID data from STM32 and data from Firebase.*

IV. Implementation process

0

- Register an account on the app or website.

01

- Logging in with an admin account involves entering the email and password of the newly registered user to activate the function of adding a new RFID.

10

- Scan the card with RFID to make parking

11

- Users can track their vehicle information and parking status on the app or website using their registered account.

100

- Scan the card with RFID to retrieve the vehicle.

V. User Interface

1. Sign-in

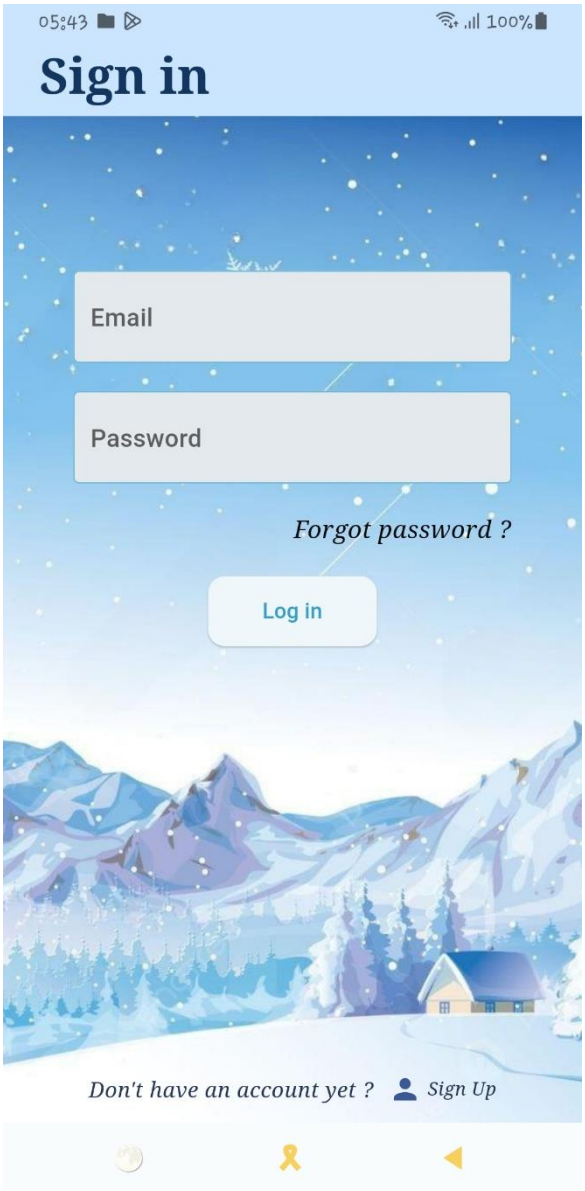


Figure 4 – Sign-in screen

Description: *Log in to the app using the registered user's email and password.*

2. Sign-up

Figure 5 - Sign-up screen

Description: *Register an account for new users*

3. Reset password

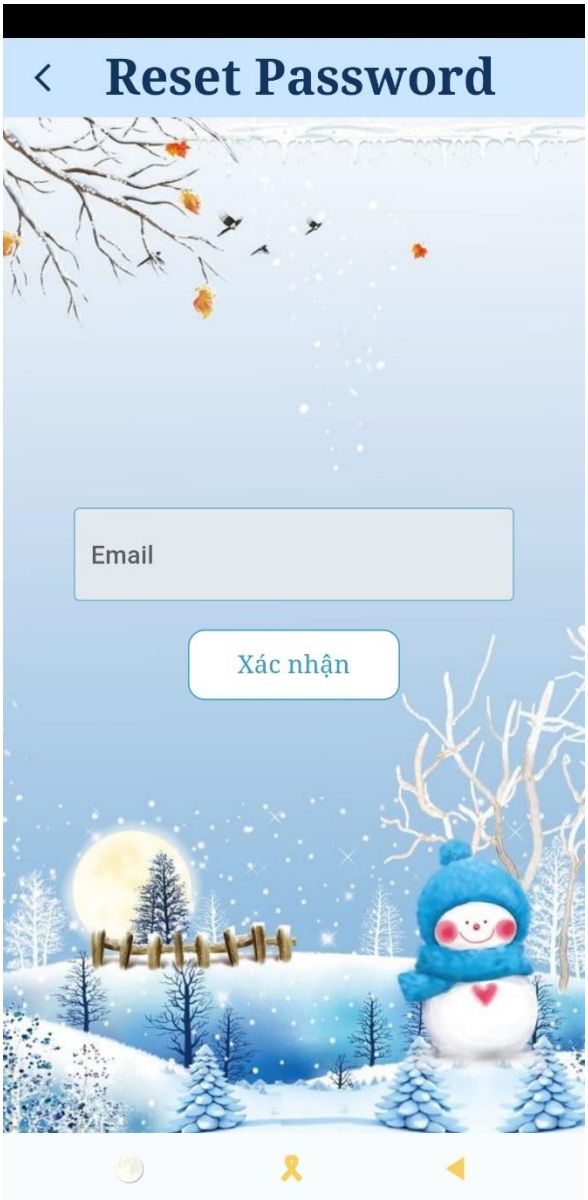


Figure 6 - Reset password screen

Description: *Users who forget their password will have to enter their registered email to reset their password.*

4. Admin

4.1 Home

Figure 7 - Admin home screen

Description: *Administrators can monitor users' vehicle parking status and daily revenue.*

4.2 Account Management

Figure 8 - User account table

Description: *Manage registered user accounts.*

4.3 Parking Management

Figure 9 - Parking management table

Description: *Table to manage the number of users sending vehicles.*

4.4 Add / Update RFID

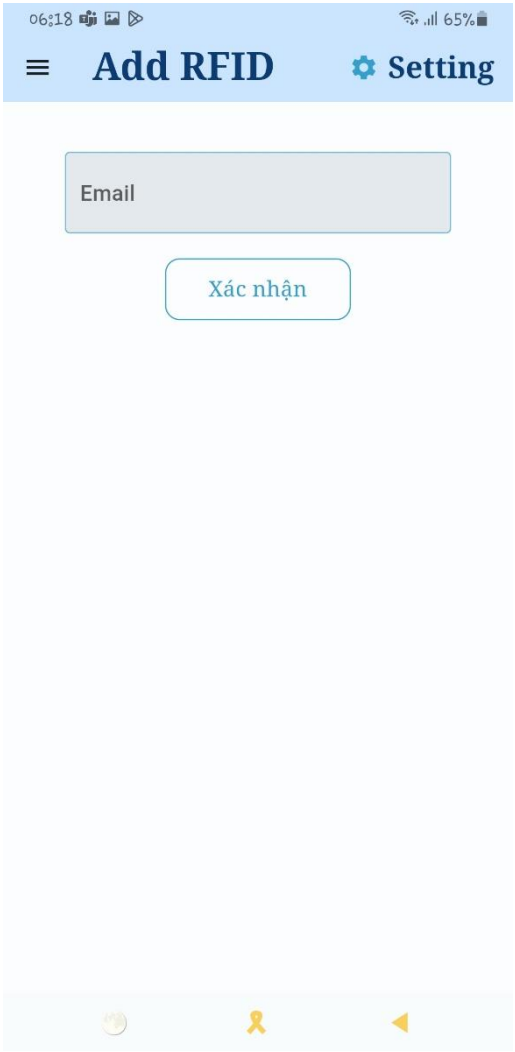


Figure 10 - Adding / Updating RFID screen

Description: *Allows administrators to add or update a user's RFID code.*

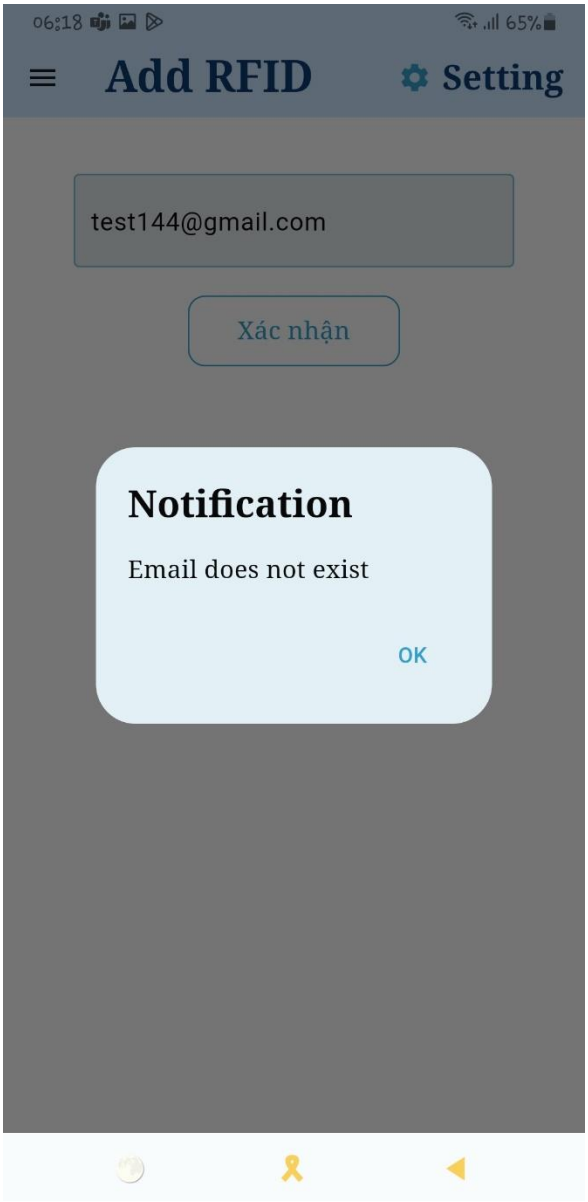
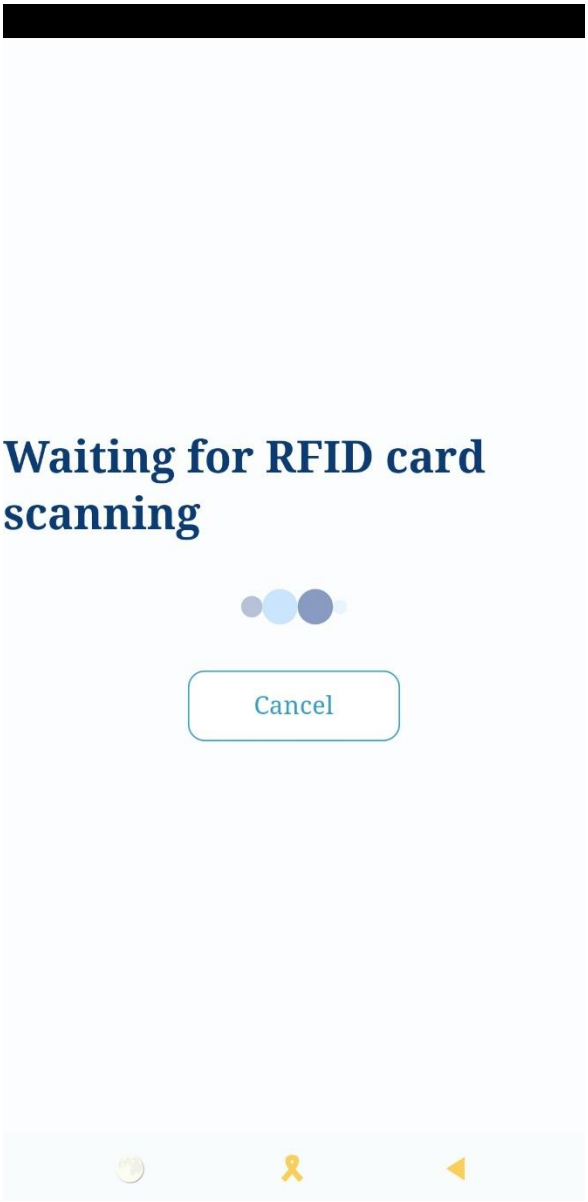


Figure 11 - Account notification does not exist Screen

Description: *Users who enter an unregistered email will display a message that the email does not exist*



Description: *User waits to scan card to register RFID code.*

4.5 Update

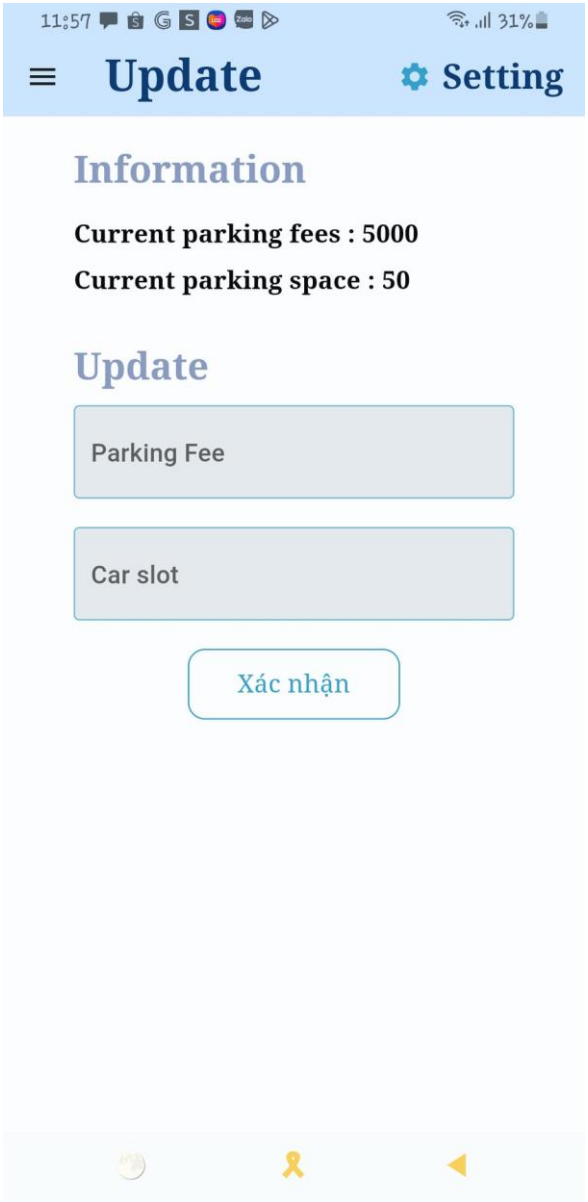


Figure 12 - Updating fare or car slot screen

Description: *Allow administrators to update the parking ticket prices or the number of parking spaces in the parking lot.*

5. Users

5.1 Home

Figure 13 - User home screen

Description: *Users can view their parking information, including user name, email, password, remaining account balance, license plate number, RFID, date and time of the last parking session, and the current status of their parked vehicle.*

5.2 Wallet

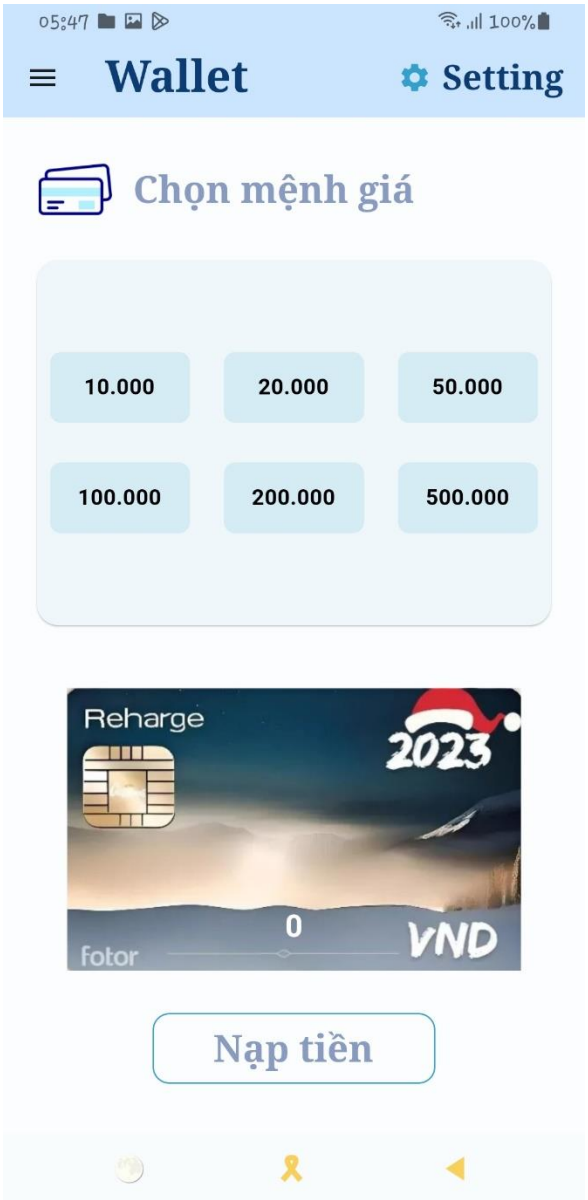


Figure 14 - Deposit money into your account screen

Description: *Users can select the desired denomination to recharge their account.*

5.3 Change user identity

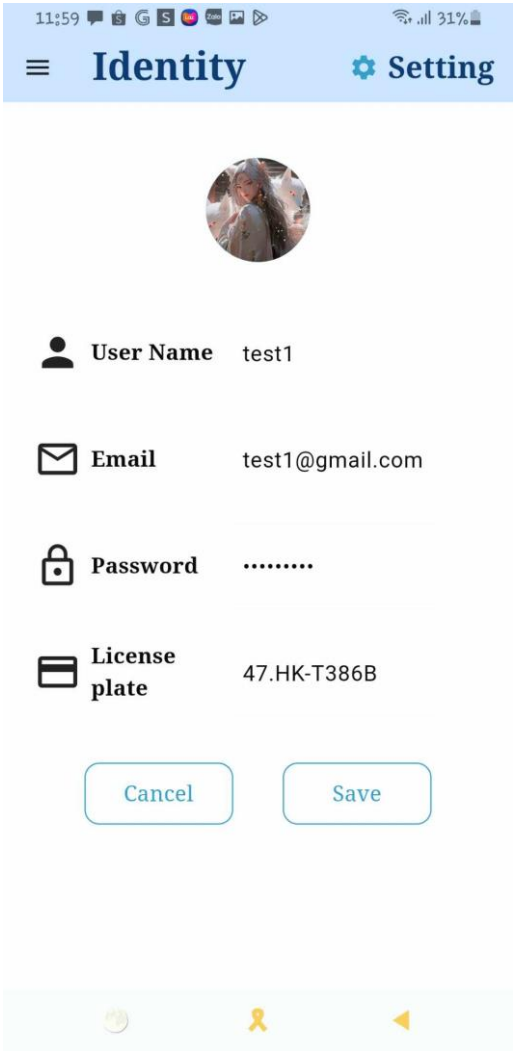


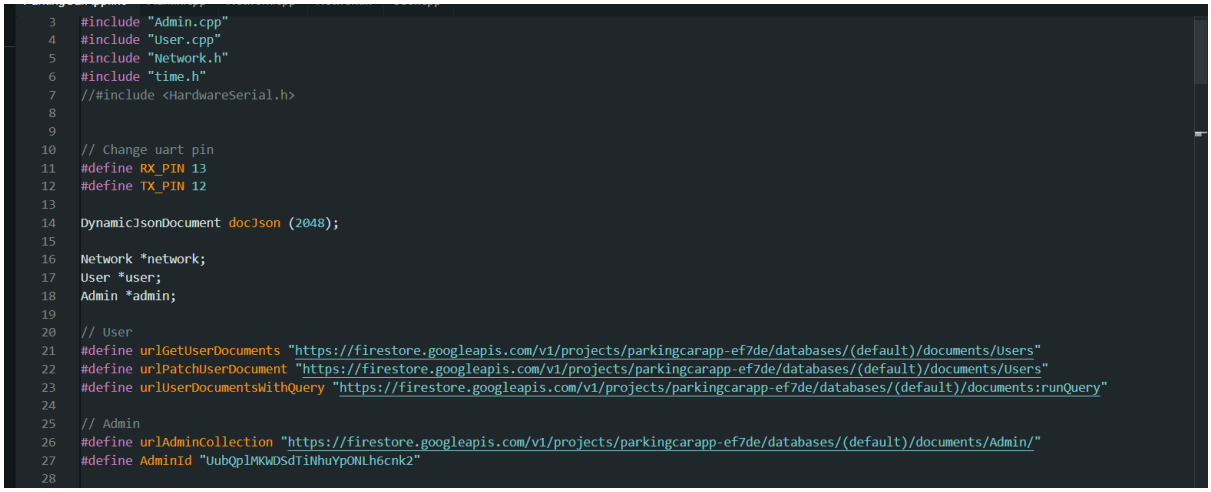
Figure 15 - Change user identity screen

Description: *Users can change their user name, email and license plate number. If the user is parking your vehicle, you will not be allowed to change the license plate.*

VI. ESP32

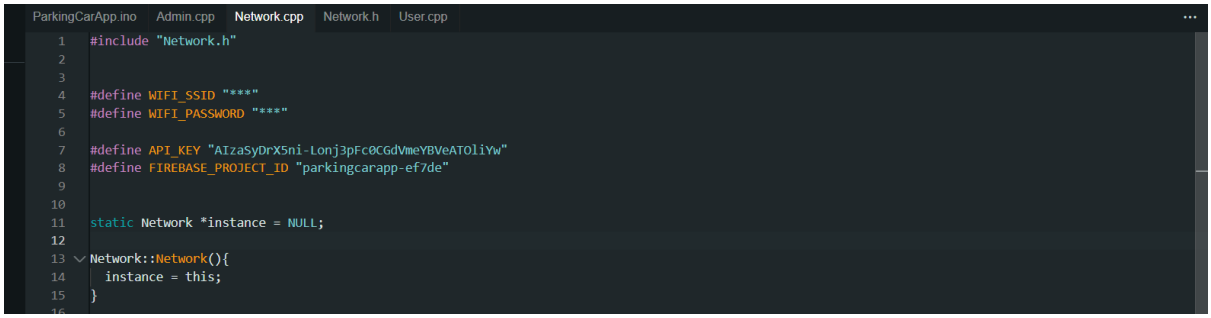
The ESP processor used in this project is ESP32-CAM Ai-Thinker.

Uart pins U0TXD, U0RXD are changed to pins GPIO13(RX) and GPIO12(TX).



```
#include "Admin.cpp"
#include "User.cpp"
#include "Network.h"
#include "time.h"
//#include <HardwareSerial.h>


// Change uart pin
#define RX_PIN 13
#define TX_PIN 12

DynamicJsonDocument docJson (2048);

Network *network;
User *user;
Admin *admin;

// User
#define urlGetUserDocuments "https://firestore.googleapis.com/v1/projects/parkingcarapp-ef7de/databases/(default)/documents/Users"
#define urlPatchUserDocument "https://firestore.googleapis.com/v1/projects/parkingcarapp-ef7de/databases/(default)/documents/Users"
#define urlUserDocumentsWithQuery "https://firestore.googleapis.com/v1/projects/parkingcarapp-ef7de/databases/(default)/documents:runQuery"

// Admin
#define urlAdminCollection "https://firestore.googleapis.com/v1/projects/parkingcarapp-ef7de/databases/(default)/documents/Admin/"
#define AdminId "UubQplMKWDSdTiNhuYpONLh6cnk2"

```

Users set up wifi password and password in the Network.cpp file



ParkingCarApp.ino | Admin.cpp | **Network.cpp** | Network.h | User.cpp

```
#include "Network.h"


#define WIFI_SSID "***"
#define WIFI_PASSWORD "***"

#define API_KEY "AIzaSyDrX5ni-Lonj3pFc0CGdVmeYBVeATOliYw"
#define FIREBASE_PROJECT_ID "parkingcarapp-ef7de"


static Network *instance = NULL;

Network::Network(){
  instance = this;
}
```

--- End ---