***DOOR LOCK AND UNLOCK SYSTEM***

#include <LiquidCrystal.h>

#include <Keypad.h>

#include <Servo.h>

#include "SafeState.h"

#include "icons.h"

/\* Locking mechanism definitions \*/

#define SERVO\_PIN 6

#define SERVO\_LOCK\_POS 20

#define SERVO\_UNLOCK\_POS 90

Servo lockServo;

/\* Display \*/

LiquidCrystal lcd(12, 11, 10, 9, 8, 7);

/\* Keypad setup \*/

const byte KEYPAD\_ROWS = 4;

const byte KEYPAD\_COLS = 4;

byte rowPins[KEYPAD\_ROWS] = {5, 4, 3, 2};

byte colPins[KEYPAD\_COLS] = {A3, A2, A1, A0};

char keys[KEYPAD\_ROWS][KEYPAD\_COLS] = {

{'1', '2', '3', 'A'},

{'4', '5', '6', 'B'},

{'7', '8', '9', 'C'},

{'\*', '0', '#', 'D'}

};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, KEYPAD\_ROWS, KEYPAD\_COLS);

/\* SafeState stores the secret code in EEPROM \*/

SafeState safeState;

void lock() {

lockServo.write(SERVO\_LOCK\_POS);

safeState.lock();

}

void unlock() {

lockServo.write(SERVO\_UNLOCK\_POS);

}

void showStartupMessage() {

lcd.setCursor(4, 0);

lcd.print("Welcome!");

delay(1000);

lcd.setCursor(0, 2);

String message = "ArduinoSafe v1.0";

for (byte i = 0; i < message.length(); i++) {

lcd.print(message[i]);

delay(100);

}

delay(500);

}

String inputSecretCode() {

lcd.setCursor(5, 1);

lcd.print("[\_\_\_\_]");

lcd.setCursor(6, 1);

String result = "";

while (result.length() < 4) {

char key = keypad.getKey();

if (key >= '0' && key <= '9') {

lcd.print('\*');

result += key;

}

}

return result;

}

void showWaitScreen(int delayMillis) {

lcd.setCursor(2, 1);

lcd.print("[..........]");

lcd.setCursor(3, 1);

for (byte i = 0; i < 10; i++) {

delay(delayMillis);

lcd.print("=");

}

}

bool setNewCode() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Enter new code:");

String newCode = inputSecretCode();

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Confirm new code");

String confirmCode = inputSecretCode();

if (newCode.equals(confirmCode)) {

safeState.setCode(newCode);

return true;

} else {

lcd.clear();

lcd.setCursor(1, 0);

lcd.print("Code mismatch");

lcd.setCursor(0, 1);

lcd.print("Safe not locked!");

delay(2000);

return false;

}

}

void showUnlockMessage() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.write(ICON\_UNLOCKED\_CHAR);

lcd.setCursor(4, 0);

lcd.print("Unlocked!");

lcd.setCursor(15, 0);

lcd.write(ICON\_UNLOCKED\_CHAR);

delay(1000);

}

void safeUnlockedLogic() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.write(ICON\_UNLOCKED\_CHAR);

lcd.setCursor(2, 0);

lcd.print(" # to lock");

lcd.setCursor(15, 0);

lcd.write(ICON\_UNLOCKED\_CHAR);

bool newCodeNeeded = true;

if (safeState.hasCode()) {

lcd.setCursor(0, 1);

lcd.print(" A = new code");

newCodeNeeded = false;

}

auto key = keypad.getKey();

while (key != 'A' && key != '#') {

key = keypad.getKey();

}

bool readyToLock = true;

if (key == 'A' || newCodeNeeded) {

readyToLock = setNewCode();

}

if (readyToLock) {

lcd.clear();

lcd.setCursor(5, 0);

lcd.write(ICON\_UNLOCKED\_CHAR);

lcd.print(" ");

lcd.write(ICON\_RIGHT\_ARROW);

lcd.print(" ");

lcd.write(ICON\_LOCKED\_CHAR);

safeState.lock();

lock();

showWaitScreen(100);

}

}

void safeLockedLogic() {

lcd.clear();

lcd.setCursor(0, 0);

lcd.write(ICON\_LOCKED\_CHAR);

lcd.print(" Safe Locked! ");

lcd.write(ICON\_LOCKED\_CHAR);

String userCode = inputSecretCode();

bool unlockedSuccessfully = safeState.unlock(userCode);

showWaitScreen(200);

if (unlockedSuccessfully) {

showUnlockMessage();

unlock();

} else {

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Access Denied!");

showWaitScreen(1000);

}

}

void setup() {

lcd.begin(16, 2);

init\_icons(lcd);

lockServo.attach(SERVO\_PIN);

/\* Make sure the physical lock is sync with the EEPROM state \*/

Serial.begin(115200);

if (safeState.locked()) {

lock();

} else {

unlock();

}

showStartupMessage();

}

void loop() {

if (safeState.locked()) {

safeLockedLogic();

} else {

safeUnlockedLogic();

}

}