Stoian Bogdan Alin -1305B

Harpa Andrei-Alexandru grupa 1305B

Iftime Adrian-Dumitru grupa 1306B



**Titlu Proiect**: Dispozitiv Smart-Gambling



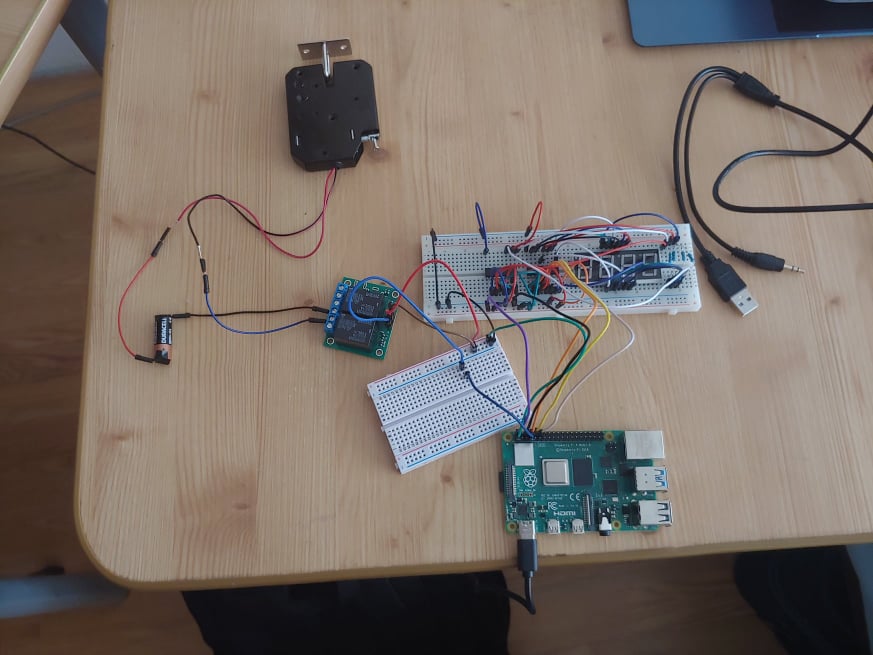
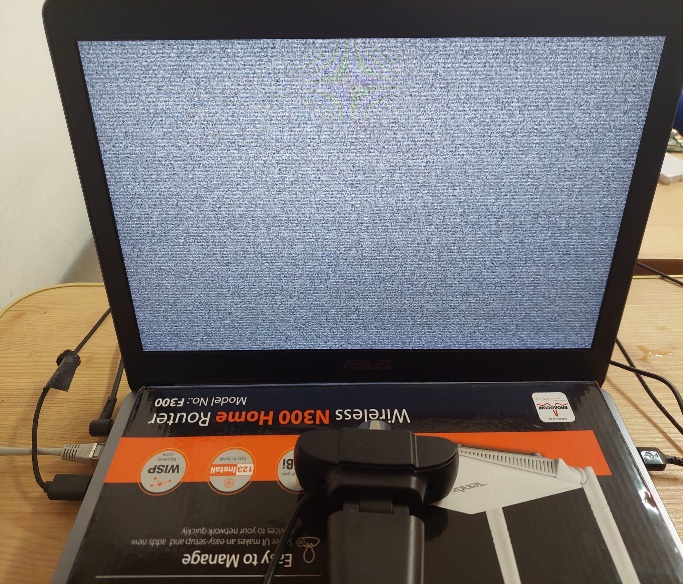
**Link demonstrație**: …..

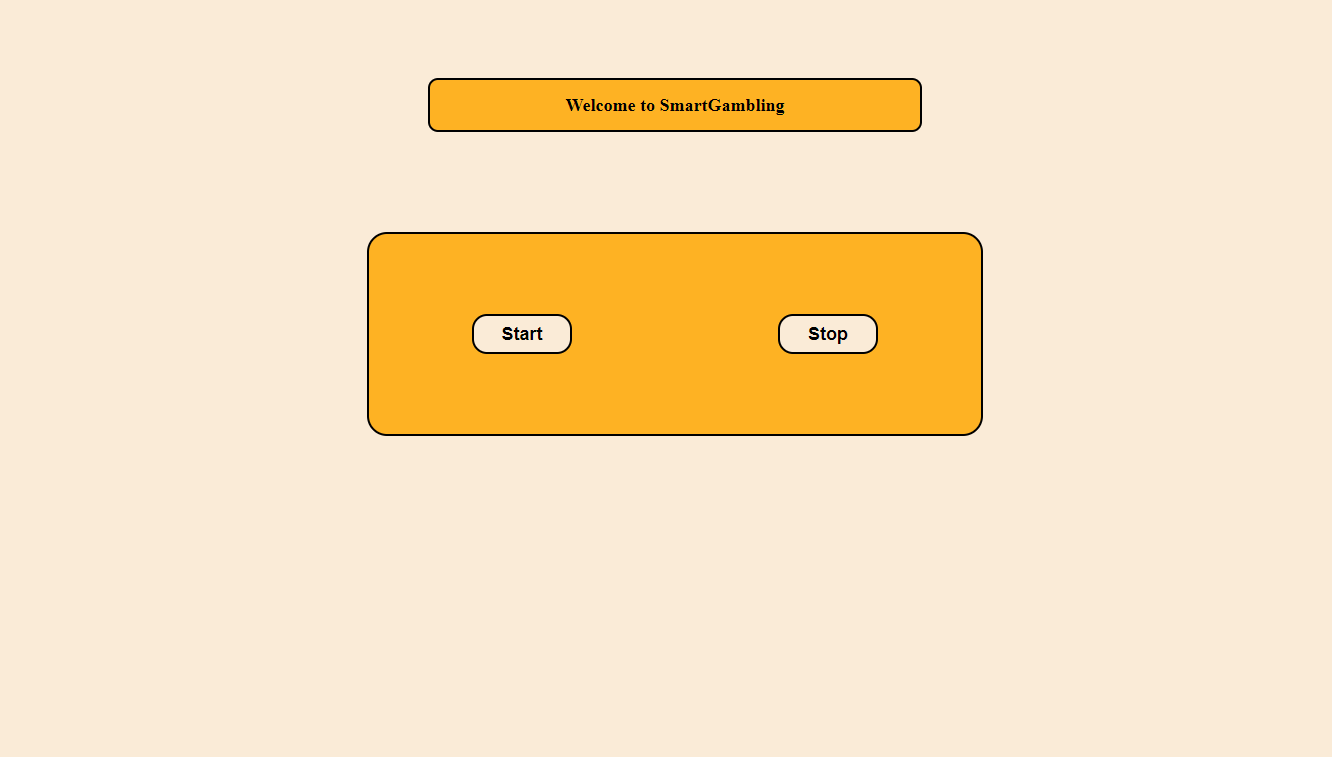
**Rezumat**

Proiectul nostru constă dintr-un dispozitiv de tipul “Joc de noroc” utilizând un generator de numere aleatoare adevarate folosind o cameră web țintită la un ecran cu zgomot “sare și piper” și o incuietoare electrică. Indiferent de rezultatul jocului de noroc, va fi trimis utilizatorului un email în care va fi menționat dacă numere lui sunt câștigătoare sau nu. Serverul va fi implementat folosind framework-ul Bottle.

**Descriere**

Odată ce serverul a fost pornit, utilizatorul dispozitivului are posibilitatea de a porni procesul de generare al numărului aleator prin apăsarea butonului de Start. După ce butonul a fost apăsat, camera web preluează o imagine, această fotografie va fi transformată într-o imagine alb-negru folosind un algoritm de thresholding. Folosind valorile de intensitate ale nuanțelor de gri ale fiecarui pixel din imagine se obține un numar mare din care noi preluăm biții 10-12, (pentru a obtine numerele mai facil) pentru a crea cate o cifră, la final având un număr de 4 cifre care va fi comparat. Această valoare va fi de asemenea afișată pe display-ul cu 7 segmente. După generarea numărului, prin intermediul librăriei smtplib se va construi corpul unui mail în care va fi menționat rezultatul tragerii la sorț care va fi trimis către contul persoanei care utilizează dispozitivul de smart-gambling. Dacă valoarea produsă de către generator este cea setată drept câștigătoare, atunci va fi trimis un semnal către releu. După primirea semnalului, releul va deschide solenoidul.

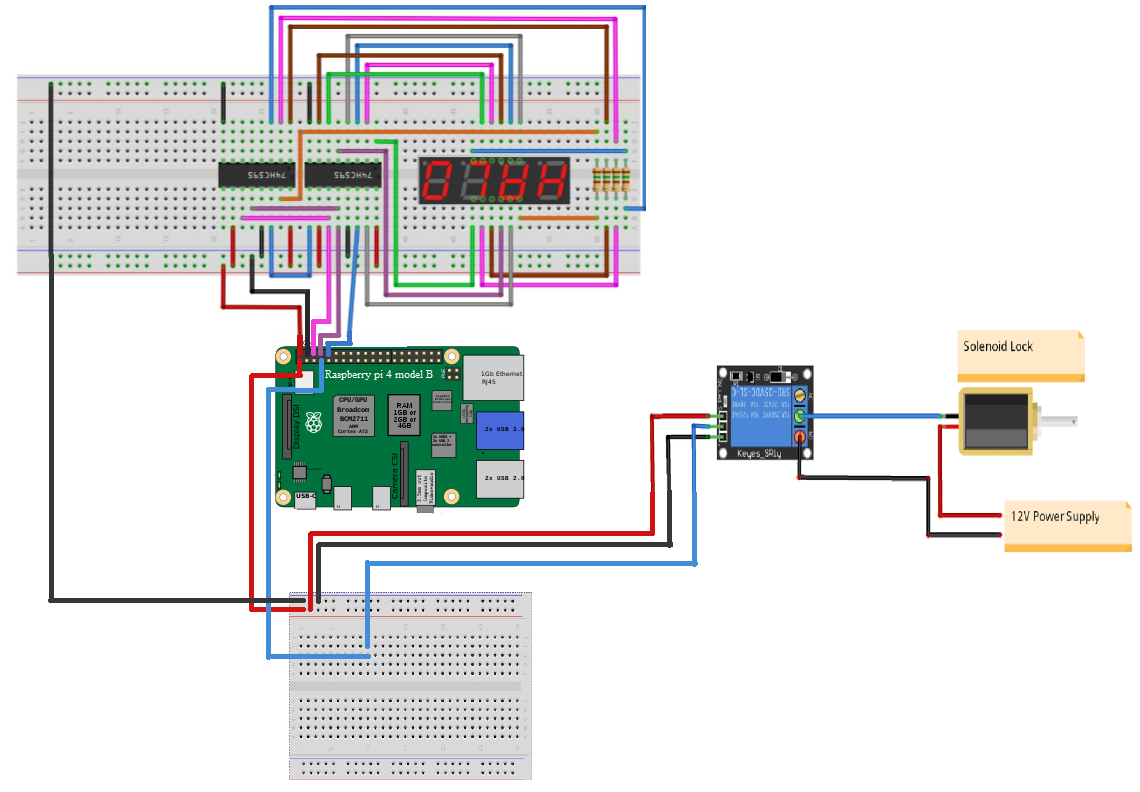
 



**Componente Hardware:**

* [Din kitul Plusivo Raspberry Pi 4:](https://www.optimusdigital.ro/ro/kituri/9555-kit-plusivo-pi-4-super-starter-cu-raspberry-pi-4-cu-2-gb-ram-si-card-noobs-de-16-gb.html?search_query=raspberry+pi+4+kit&results=65)
* Raspberry Pi 4 cu memorie de 2 GB x1
* Fire Male/Female x10
* Fire Male/Male x29
* Rezistoare 150 ohm x4
* Display 7-segmente si 4 cifre, cu catod comun x1
* Registre shiftare 74HC595 x2
* [Releu Pololu Basic 2-Channel SPDT Carrier cu 5VDC](https://www.pololu.com/product/2485/specs) x1
* [Yala electromagnetică blocaj electronic solenoid 12V, 2A](https://www.emag.ro/yala-electromagnetica-blocaj-electronic-solenoid-inchidere-usa-dulap-seif-12v-2a-cl-601a/pd/DQQRCWBBM/) x1
* Baterie Duracel 12v 23A x1
* [Camera Web PC](https://www.emag.ro/camera-web-csc-720p-neagra-59021/pd/DF8627MBM/) x1

**Schematica circuitului:**



**Software:**

Pe partea de software am instalat pe raspberry pi sistemul de operare Raspberry Pi Raspbian.

Pe lângă sistemul de operare, au fost folosite librăriile următoare:

* PIL
* email
* subprocess
* smtplib
* RPi
* time
* sys
* bottle

Pentru pornirea serverului folosind un calculator la distantă s-a folosit aplicația Putty.

**Story:**

Cu toții visăm să obținem sume de bani usor fără a face prea multă muncă. Pentru a veni în ajutorul persoanelor care au acest plan în minte, am creat un dispozitiv de smart gambling. Acest aparat, poate să ii facă bogați și pe cei cărora le surâde norocul dar mai ales pe proprietarii dispozitivului. Fiind un dispozitiv de tipul „Joc de Noroc”, odată ce utilizatorul câstigă, cealaltă persoană (proprietarul aparatului) iese in pierdere. Faptul că interfata cu utilizatorul se face cu ajutorul paginii web este un plus pentru că oferă modalități speciale de amplasare a ușii cu incuietoare. Un astfel de exemplu ar putea fi: Odată ce numerele câstigătoare au fost obținute, se deschide o usă din tavan si banii cad direct peste utilizatorul aparatului.

**Cod:**

**Algoritm.py**

from PIL import Image as Image

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

import subprocess

import smtplib

import RPi.GPIO as GPIO

import time

import sys

GPIO.setmode(GPIO.BOARD)

dataPin = 12

latchPin = 10

clockPin = 8

GPIO.setup(dataPin, GPIO.OUT)

GPIO.setup(latchPin, GPIO.OUT)

GPIO.setup(clockPin, GPIO.OUT)

GPIO.setup(7, GPIO.OUT)

GPIO.output(dataPin, GPIO.LOW)

GPIO.output(latchPin, GPIO.LOW)

GPIO.output(clockPin, GPIO.LOW)

def send\_mail(mail\_content):

#adresa de mail si parola proprietarului dispozitivului

sender\_address = 'stoiam.labsm@gmail.com'

sender\_pass = 'PROIECTSM'

# adresa de mail a utilizatorului

receiver\_address = 'stoiam.labsm@gmail.com'

#construirea antentului mesajului

message = MIMEMultipart()

message['From'] = sender\_address

message['To'] = receiver\_address

message['Subject'] = 'SmartGambling'

#selectarea tipului mesajului

message.attach(MIMEText(mail\_content, 'plain'))

#conectarea la serverul de gmail prin intermediul librariei smtplib si a credentialelor proprietarului dispozitivului

session = smtplib.SMTP('smtp.gmail.com', 587)

session.starttls() #activarea securitatii

session.login(sender\_address, sender\_pass)

text = message.as\_string()

#trimiterea propriu zisa a mesajului odata ce acesta a fost construit

session.sendmail(sender\_address, receiver\_address, text)

session.quit()

print('Mail Sent')

#functie care seteaza ce cifra sa fie afisata pe display

def Digit(x):

global digit

if x == 1:

digit = 14

elif x == 2:

digit = 13

elif x == 3:

digit = 11

elif x == 4:

digit = 7

#functie care se ocupa de shiftarea si afisarea cifrelor in functie de digitul mentionat inainte si bufferul trimis ca parametru

def shift(buffer):

global digit

for i in range(0,8):

GPIO.output(dataPin, (128 & (digit<<i)))

GPIO.output(clockPin, GPIO.HIGH)

time.sleep(0.001)

GPIO.output(clockPin, GPIO.LOW)

for i in range(0,8):

GPIO.output(dataPin, (128 & (buffer << i)))

GPIO.output(clockPin, GPIO.HIGH)

time.sleep(0.001)

GPIO.output(clockPin, GPIO.LOW)

GPIO.output(latchPin, GPIO.HIGH)

time.sleep(0.001)

GPIO.output(latchPin, GPIO.LOW)

name = 0

nr = 4

devNull = open('/dev/null', 'w')

iter = 0

digit1 = 0

digit2 = 0

digit3 = 0

digit4 = 0

#functia care genereaza numere aleatorii folosindu-se de poza obtinuta de camera web

def getNumbers():

global name

global nr

global devNull

global iter

global digit1

global digit2

global digit3

global digit4

while(iter < nr):

name = name + 1

randomBits = ""

pixelRow = 0

pixelColumn = 0

captureImage = subprocess.Popen(["fswebcam", "-r", "356x292", "-d", "/dev/video0", "static.jpg", "--skip", "10", "--set", "brightness=50%", "-D", "3"], stdout=devNull, stderr=devNull)

#instructiune care are ca scop obtinerea unei fotografii prin intermediul camerei web

captureImage.communicate()

#deschiderea imaginii

staticImage = Image.open("static.jpg")

staticImage = staticImage.crop((1,1,50,50))

#transformarea fotografiei color in una alb-negru

bW\_Image = staticImage.convert('1')

#salvarea informatiilor din imagine intr-o variabila care poate fi parcursa

imageToProcess = bW\_Image.load()

#parcurgerea variabilei pentruj a obtine valorile numarului aleator

while pixelRow < staticImage.size[0]:

while pixelColumn < staticImage.size[1]:

if imageToProcess[pixelRow, pixelColumn] == 0:

#adaugarea unui 0 valorii aleatoare daca pixelul curent este alb

randomBits = randomBits + "0"

else:

#adaugarea unui 1 valorii aleatoare daca numarul curent este negru

randomBits = randomBits + "1"

pixelColumn = pixelColumn + 1

pixelRow = pixelRow + 1

pixelColumn = 0

output = open('output.txt', 'w')

#scrierea numarului generat intr-un fisier pentru a putea fi obtinut in caz de nevoie

output.write(randomBits[10:13])

#afisarea numarului generat in terminal

print (randomBits[10:13])

if(iter == 0):

digit1 = randomBits[10:13]

elif(iter == 1):

digit2 = randomBits[10:13]

elif(iter == 2):

digit3 = randomBits[10:13]

elif(iter == 3):

digit4 = randomBits[10:13]

output.close()

iter = iter + 1

def main():

#apelul functiei care genereaza cele 4 numere aleatorii

getNumbers()

dictionary = {"000": 63, "001": 6, "010": 91, "011": 79, "100": 102, "101": 109, "110": 125, "111": 7}

#salvarea numerelor in variabile locale

digit1Local = dictionary[digit1]

digit2Local = dictionary[digit2]

digit3Local = dictionary[digit3]

digit4Local = dictionary[digit4]

#daca toate numerele generate sunt egale, trimite semnal catre releu pentru a deschide incuietoarea si trimite mesajul catre mail-ul castigatorului

if digit1 == digit2 and digit3 == digit4 and digit1 == digit3:

GPIO.output(7, GPIO.HIGH)

send\_mail("Felicitari, ati castigat! Numerele dumneavoastra sunt: " + str(int(digit1, base=2)) + ", " + str(int(digit2, base=2)) + ", " + str(int(digit3, base=2)) + ", " + str(int(digit4, base=2)) + ". Sa aveti o zi minunata!")

time.sleep(8)

GPIO.output(7, GPIO.LOW)

#numerele aleatoare nu sunt cele castigatoare =>trimite mesajul pe email-ul userului

else:

send\_mail("Din pacate nu ati castigat de aceasta data, numerele dumneavoastra sunt: " + str(int(digit1, base=2)) + ", " + str(int(digit2, base=2)) + ", " + str(int(digit3, base=2)) + ", " + str(int(digit4, base=2)) + ". Va rugam sa mai incercati. O zi buna!")

#afiseaza pe display-ul cu 7 segmente numerele obtinute cate o cifra pe rand cu viteza foarte mare astfel incat sa para ca sunt afisate toate deodata

while True:

try:

#setare cifra 1 si afisarea ei

Digit(1)

shift(digit1Local)

#setare cifra 2 si afisarea ei

Digit(2)

shift(digit2Local)

#setare cifra 3 si afisarea ei

Digit(3)

shift(digit3Local)

#setare cifra 4 si afisarea ei

Digit(4)

shift(digit4Local)

except KeyboardInterrupt:

GPIO.cleanup()

print("done")

sys.exit()

if \_\_name\_\_ == "\_\_main\_\_":

main()

**server.py**

from bottle import Bottle, route, run, template, request, redirect

import Algoritm

import subprocess

app = Bottle()

@app.route('/')

#ruta pentru pagina de home din care utilizatorul poate da Start

def index():

return template('index.html')

project\_process = None

#ruta accesata la apasarea butonului start

@app.route('/proiect', method="POST")

def run\_proiect():

global project\_process

#se creeaza un nou subproces care are ca scop rularea algoritmului de generare a numarului aleator

project\_process = subprocess.Popen(["python", "proiect.py"])

redirect('/')

#ruta accesata odata ce numerele aleatoare au fost generate si rezultatul a fost trimis catre utilizator

@app.route('/termina\_proiect', method="POST")

def terminate\_project():

global project\_process

if(project\_process != None):

project\_process.terminate()

project\_process = None

redirect('/')

run(app, host='0.0.0.0', port='8080', debug=True)

**index.html**

<!DOCTYPE HTML>

<html>

<head>

<title>Proiect SM</title>

<style>

body {

background-color: antiquewhite;

}

main {

margin-left: 35%;

margin-top: 5%;

}

.controls {

display: flex;

justify-content: space-around;

align-items: center;

border: 2px solid black;

border-radius: 20px;

width: 50%;

height: 200px;

background-color: rgba(255, 166, 0, 0.836);

}

#start, #stop {

margin-top: 30px;

margin-bottom: 30px;

}

#startB, #stopB {

background-color: antiquewhite;

width: 100px;

height: 40px;

border-radius: 15px;

border: 2px solid black;

font-size: large;

font-weight: bold;

}

.titlu {

width: 40%;

height: 50px;

font-size: large;

font-weight: bold;

background-color: rgba(255, 166,0 , 0.836);

border-radius: 10px;

margin-bottom: 100px;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

border: 2px solid black;

margin-left: 5%;

}

</style>

</head>

<body>

<main>

<div class="titlu">

Welcome to SmartGambling

</div>

<div class="controls">

<!—formular cu butonul care porneste procesul de generare al numerelor-->

<form method="POST" action="/proiect" id="start">

<input type=submit name="proiect" value="Start" id="startB"/>

</form>

<!—formular cu butonul care opreste procesul de generare al numerelor-->

<form method="POST" action="/termina\_proiect" id="stop">

<input type=submit name="termina" value="Stop" id="stopB"/>

</form>

</div>

</main>

</body>

</html>