LAMPIRAN

```
import cv2
import argparse
import datetime
import pyzbar.pyzbar as pyzbar


ap = argparse.ArgumentParser()
ap.add_argument("-o", "--output", type=str, default= "in.csv",
                help="menyimpan waktu kedatangan dalam .CSV")
args = vars(ap.parse_args())

cap = cv2.VideoCapture(0)
CSV = open(args["output"],"w")
gett = set()
i=0

while True:
    _, frame = cap.read()
    deObjects = pyzbar.decode(frame)

    for qrCode in deObjects:
        (x, y, w, h) = qrCode.rect
        cv2.rectangle(frame, (x, y), (x + w, y + h),(0, 0, 255),2)

        i=i+1
        cv2.imshow('guest', frame)
        cv2.imwrite('image/guest.' + str(i) + '.png', frame)
        cv2.waitKey(5000)
        qrData = qrCode.data.decode("utf-8")
        qrType = qrCode.type

        text = "{} ({})".format(qrData, qrType)
        cv2.putText(frame, text, (x, y - 10),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

        if qrData not in gett:
            CSV.write("{},{}\n".format(datetime.datetime.now(),
                      qrData))

    cv2.imshow("Frame", frame)
    key = cv2.waitKey(1)

    if key == ord('q'):
        break
    if key == ord('c'):
        i = i+1
        cv2.imshow("imshow2", frame)
        cv2.imwrite('image/guest.' + str (i) + '.png', frame)
        print("Image Captured")

cap.release()
cv2.destroyAllWindows()
```

```
from tkinter import *
import time
from basedata import Database
from tkinter import messagebox
import argparse
import datetime

ap = argparse.ArgumentParser()
ap.add_argument("-o", "--output", type=str, default= "out.csv",
                help="menyimpan waktu kepergian dalam .CSV")
args = vars(ap.parse_args())
CSV = open(args["output"],"w")
get = set()

database = Database("guest.db")

class Window(object):
    def __init__(self,window):
        self.window = window
        self.window.wm_title("SISTEM_RFID")

        l1 = Label(window, text="Tap Kartu Anda Disini",
                   font="Times 18", fg="#000")
        l1.place(x=109, y=78)
        l1.config(bg="#fff")

        self.rfid_tap = StringVar()
        self.eb1 = Entry(window, textvariable=self.rfid_tap,
                         width = 40)
        self.eb1.place(x=89, y=120)
        self.eb1.config(bg="#fff")
        self.eb1.bind("<Return>", self.LoadData)

        b1 = Button(window, text="Enter", width=12, height = 2,
                    bd= 0, bg="#42f498", activebackground="#fff",
                    activeforeground="#42f498", fg="#fff",
                    command=self.LoadData)
        b1.place(x=164, y=150)
        b1.bind("<Button-1>",self.LoadData)

        self.extention()

    def extention(self):
        ex1 = Label(window, text="Nama")
        ex1.place(x=420, y=80)
        ex1.config(bg="#fff")

        self.ex1a = Listbox(window, height=2, width=40)
        self.ex1a.place(x=570, y=72)
        self.ex1a.config(bg="#fff")

        ex2 = Label(window, text="Nomer Kendaraan")
        ex2.place(x=420, y=140)
        ex2.config(bg="#fff")

        self.ex2a = Listbox(window, height=2, width=40)
        self.ex2a.place(x=570, y=132)
        self.ex2a.config(bg="#fff")

        ex3 = Label(window, text="Jenis Kendaraan")
        ex3.place(x=420, y=200)
        ex3.config(bg="#fff")
        self.ex3a = Listbox(window, height=2, width=40)
        self.ex3a.place(x=570, y=192)
        self.ex3a.config(bg="#fff")

    def LoadData(self, event):
        tag_tap = self.rfid_tap.get()
        self.ex1a.delete(0, END)
        self.ex2a.delete(0, END)
        self.ex3a.delete(0, END)
        for row in database.tapname(self.rfid_tap.get()):
            self.ex1a.insert(END, row)
        for row in database.tapplat(self.rfid_tap.get()):
            self.ex2a.insert(END, row)
        for row in database.taproda(self.rfid_tap.get()):
            self.ex3a.insert(END, row)

        if not tag_tap:
            messagebox.showwarning("PERINGATAN","TOLONG TAP KARTU
                                   ANDA KEMBALI !")
            return False

        if not database.tapname(self.rfid_tap.get()):
            messagebox.showwarning("PERINGATAN", "MAAF KARTU ANDA
                                   TIDAK TERDAFTAR \nANDA TIDAK
                                   DAPAT MENINGGALKAN LOKASI
                                   !\nMOHON HUBUNGI PETUGAS")
            self.eb1.delete(0, END)
            return False

        if tag_tap not in get:
            CSV.write("{},{}\n".format(datetime.datetime.now(),
                      database.tap_all(self.rfid_tap.get())))
            return False

        time.sleep(2)
        print("Resset")
        self.eb1.delete(0, END)

window = Tk()
window.geometry('900x360+300+200')
window.config(bg="#fff")
Window(window)

window.mainloop()
```

```
from tkinter import *
from Finalization.basedata import Database
import pyautogui
import time

database = Database('guest.db')

class Window(object):
    def __init__(self,window):
        self.window = window
        self.window.countings = 0
        self.window.wm_title("SISTEM COUNTING")

        l1 = Label(window, text="Jumlah kendaraan masuk : ",
                   font="Times 18", fg="#000")
        l1.place(x=59, y=58)
        l1.config(bg="#fff")

        l2 = Label(window, text="Jumlah kendaraan keluar : ",
                   font="Times 18", fg="#000")
        l2.place(x=59, y=128)
        l2.config(bg="#fff")

        self.variable = 20
        self.reset = self.variable = 20
        self.L = Label (window, text=self.variable, font="Times
                        18", fg="#000")
        self.L.place(x=59, y=10)
        self.L.config(bg="#fff")

        self.masuk_text = IntVar()
        self.ex1a = Entry(window, textvariable=self.masuk_text)
        self.ex1a.place(x=326, y=58)
        self.ex1a.config(bg="#fff")

        self.keluar_text = IntVar()
        self.ex2a = Entry(window, textvariable=self.keluar_text)
        self.ex2a.place(x=326, y=128)
        self.ex2a.config(bg="#fff")
        self.ex2a.bind("<Return>", self.masuk_keluar)

        self.tombol = Button(window, text='',width=12, height = 2,
                             command=self.keluar_masuk)
        self.tombol.place(x=249, y=302)
        self.tombol.config(bd=0, bg="#fff",
                           activebackground="#fff",
                           activeforeground="#fff", fg="#fff")

        self.tombo2 = Button(window, text='Lihat semua', width=12,
                             height=2, command=self.masuk_keluar)
        self.tombo2.place(x=249, y=202)
        self.tombo2.config(bd=0, bg="#42f498",
                           activebackground="#fff",
                           activeforeground="#42f498", fg="#fff")
        self.tombo2.bind("<Button-1>", self.masuk_keluar)

        self.tombo3 = Button(window, text='Reset', width=12,
                             height=2, command=self.reset_sistem)
        self.tombo3.place(x=359, y=202)
        self.tombo3.config(bd=0, bg="#42f498",
                           activebackground="#fff",
                           activeforeground="#42f498", fg="#fff")
    def lihat_semua(self):
        self.liat=Toplevel()
        self.liat.title("Monitoring Kendaraan")
        self.liat.geometry("415x450+120+120")
        l_1 = Label(self.liat, text="Kendaraan Masuk : ",
                    font="Times 12", fg="#000")
        l_1.grid(row=0, column=0)

        l_2 = Label(self.liat, text="Kendaraan Keluar : ",
                    font="Times 12", fg="#000")
        l_2.grid(row=12, column=0)

        self.list_1 = Listbox(self.liat, height=11, width=60)
        self.list_1.grid(row=2, column=0, rowspan=6, columnspan=2)

        self.list_2 = Listbox(self.liat, height=11, width=60)
        self.list_2.grid(row=13, column=0, rowspan=6,
                         columnspan=2)

        sb1 = Scrollbar(self.liat)
        sb1.grid(row=2, column=2, rowspan=6)
        self.list_1.config(yscrollcommand=sb1.set)
        sb1.config(command=self.list_1.yview)

        sb2 = Scrollbar(self.liat)
        sb2.grid(row=13, column=2, rowspan=6)
        self.list_1.config(yscrollcommand=sb2.set)
        sb2.config(command=self.list_2.yview)

        tb1 = Button(self.liat, text="Lihat Semua", )
        tb1.place(x=141, y=414)

    def keluar_masuk(self, *args):
        self.ex1a.delete(0, END)
        self.ex2a.delete(0, END)
        for row in database.kendaraan_masuk():
            self.ex1a.insert(END, row)
        for row in database.kendaraan_keluar():
            self.ex2a.insert(END, row)
        self.variable += self.keluar_text.get()
        self.variable -= self.masuk_text.get()
        self.L['text'] = 'Jumlah Sisa Parkir Kendaraan: ' +
                          str(self.variable)

    def masuk_keluar(self, event):
        self.ex1a.delete(0, END)
        self.ex2a.delete(0, END)
        self.variable = self.reset
        self.L['text'] = 'Jumlah Sisa Parkir Kendaraan: ' +
                         str(self.variable)
        self.tombol.invoke()
        pyautogui.press('enter')

    def reset_sistem(self):
        database.reset_keluar()
        database.reset_masuk()

window = Tk()
window.geometry('600x290+300+200')
window.config(bg="#fff")
Window(window)

window.mainloop()
```

```
import sqlite3

        self.conn = sqlite3.connect(“guest.db”)
        self.cur = self.conn.cursor()
        self.cur.execute("CREATE TABLE IF NOT EXISTS guest(id
                         INTEGER PRIMARY KEY, nama TEXT, nocard
                         TEXT, noplat TEXT, roda TEXT)")
        self.conn.commit()
```

```
import sqlite3

        self.conn = sqlite3.connect(“guest.db”)
        self.cur = self.conn.cursor()
        self.cur.execute("INSERT INTO guest VALUES
                       (1,arief,0008405126,AB1234JU,roda empat),
                       (2,isan,0008833585,AB1234JU,roda dua),
                       (3,barzein,0008396787,AB1234JU,roda dua),
                       (4,salman,0008831277,AB1234JU,roda empat),
                       (5,reza,0008820408,AB1234JU,roda dua)
                        ")
        self.conn.commit()
```

```
import sqlite3

class Database:
    def __init__(self,db):
        self.conn = sqlite3.connect(db)
        self.cur = self.conn.cursor()
        self.cur.execute("CREATE TABLE IF NOT EXISTS guest(id
                         INTEGER PRIMARY KEY, nama TEXT, nocard
                         TEXT, noplat TEXT, roda TEXT)")
        self.conn.commit()
    def insert(self, nama, nocard, noplat, roda):
        self.cur.execute("INSERT INTO guest VALUES(NULL,?,?,?,?)",
                        (nama,nocard,noplat, roda))
        self.conn.commit()
    def view(self):
        self.cur.execute("SELECT * FROM guest")
        rows = self.cur.fetchall()

        return rows
    def search(self, nama='', nocard='', noplat='', roda=''):
        self.cur.execute("SELECT * FROM guest WHERE (nama = ? OR
                         nocard = ? OR noplat = ? OR roda = ?)",
                         (nama, nocard, noplat, roda))
        rows = self.cur.fetchall()
        return rows
    def tap_all(self, nocard=''):
        self.cur.execute("SELECT nama, nocard, noplat, roda  FROM
                         guest WHERE (nocard = ?)",
                         (nocard,))
        rows = self.cur.fetchall()
        return rows
    def tapname(self, nocard=''):
        self.cur.execute("SELECT nama FROM guest WHERE (nocard =
                         ?)", (nocard, ))
        rows = self.cur.fetchall()
        return rows

    def tapplat(self, nocard=''):
        self.cur.execute("SELECT noplat FROM guest WHERE (nocard =
                         ?)", (nocard, ))
        rows = self.cur.fetchall()
        return rows

    def taproda(self, nocard=''):
        self.cur.execute("SELECT roda FROM guest WHERE (nocard =
                         ?)", (nocard, ))
        rows = self.cur.fetchall()
        return rows

    def delete(self,id):
        self.cur.execute("DELETE FROM guest WHERE id = ?", (id,))
        self.conn.commit()
        #conn.close()

    def update(self,id, nama, nocard, noplat, roda):
        self.cur.execute("UPDATE guest SET nama = ?, nocard = ?,
                         noplat = ?, roda = ? WHERE id = ?",
                         (nama, nocard, noplat, roda, id))
        self.conn.commit()

    def __del__(self):
        self.conn.close()
```

```
from tkinter import *
from tkinter import messagebox
from basedata import Database

database = Database("guest.db")

class Window(object):
    def __init__(self,window):
        self.window = window
        self.window.wm_title("SISTEM DATABASE")

        tombol_daftar = Button(window,text="Daftar Baru",width=12,
                               height= 2,command=self.daftar_data)
        tombol_daftar.place(x=170, y=75)
        tombol_daftar.config(bd=0, bg="#42f498", activebackground=
                             "#fff", activeforeground="#42f498",
                             fg="#fff")

        tombol_hapus = Button(window, text="Hapus Data", width=12,
                              height = 2, command=self.hapus_data)
        tombol_hapus.place(x=55, y=150)
        tombol_hapus.config(bd=0, bg="#42f498", activebackground=
                            "#fff", activeforeground="#42f498",
                            fg="#fff")

        tombol_perbarui = Button(window, text="Perbarui Data",
                                 width=12, height = 2,
                                 command=self.perbarui_data)
        tombol_perbarui.place(x=170, y=150)
        tombol_perbarui.config(bd=0, bg="#42f498", activeback -
                               ground="#fff", activeforeground=
                               "#42f498", fg="#fff")

        tombol_data = Button(window, text="Show All Data",
                             width=12, height = 2,
                             command=self.all_database)
        tombol_data.place(x=55, y=75)
        tombol_data.config(bd=0, bg="#42f498",
                           activebackground="#fff",
                           activeforeground="#42f498", fg="#fff")


    def daftar_data(self):
        self.daftar = Toplevel()
        self.daftar.title("Daftar Baru")
        self.daftar.geometry("700x300+300+20")

        label_nama = Label(self.daftar, text="Nama")
        label_nama.place(x=30, y=35)
        l1 = Label(self.daftar, text=":")
        l1.place(x=150, y=35)

        label_nocard = Label(self.daftar, text="Nomer Kartu")
        label_nocard.place(x=30, y=70)
        l2 = Label(self.daftar, text=":")
        l2.place(x=150, y=70)

        label_noplat = Label(self.daftar, text="Nomer Kendaraan")
        label_noplat.place(x=30, y=105)
        l3 = Label(self.daftar, text=":")
        l3.place(x=150, y=105)

        label_roda = Label(self.daftar, text="Jenis Kendaraan")
        label_roda.place(x=30, y=140)
        l4 = Label(self.daftar, text=":")
        l4.place(x=150, y=140)

        label_data = Label(self.daftar, text="List Database")
        label_data.place(x=370, y=15)

        self.nama_text = StringVar()
        self.e1 = Entry(self.daftar, textvariable=self.nama_text)
        self.e1.place(x=190, y=35)

        self.nocard_text = StringVar()
        self.e2 = Entry(self.daftar,textvariable=self.nocard_text)
        self.e2.place(x=190, y=70)

        self.noplat_text = StringVar()
        self.e3 = Entry(self.daftar,textvariable=self.noplat_text)
        self.e3.place(x=190, y=105)

        self.roda_text = StringVar()
        self.e4 = Entry(self.daftar, textvariable=self.roda_text)
        self.e4.place(x=190, y=140)

        b1 = Button(self.daftar, text="Enter", width=12, height=2,
                    command=self.message_input)
        b1.place(x=164, y=190)

        self.list1 = Listbox(self.daftar, height=12, width=45)
        self.list1.place(x=370, y=35)

    def message_input(self):
        nama = self.nama_text.get()
        nocard = self.nocard_text.get()
        noplat = self.noplat_text.get()
        roda = self.roda_text.get()
        if not nama:
            messagebox.showwarning("Input kosong", "Tolong
                                   lengkapi data pada kolom
                                   kosong.")
            return False
        if not nocard:
            messagebox.showwarning("Input kosong", "Tolong
                                   lengkapi data pada kolom
                                   kosong.")
            return False
        if not noplat:
            messagebox.showwarning("Input kosong", "Tolong
                                   lengkapi data pada kolom
                                   kosong.")
            return False
        if not roda:
            messagebox.showwarning("Input kosong", "Tolong
                                   lengkapi data pada kolom
                                   kosong.")
            return False
        if nama + noplat + noplat + roda:
            self.list1.delete(0, END)
            database.insert(self.nama_text.get(),
                            self.nocard_text.get(),
                            self.noplat_text.get(),
                            self.roda_text.get())
            self.list1.delete(0, END)
            self.list1.insert(END, (self.nama_text.get(),
                                    self.nocard_text.get(),
                                    self.noplat_text.get(),
                                    self.roda_text.get()))
            messagebox.showinfo("Input berhasil","Data telah
                                berhasil terekam")


    def hapus_data(self):
        self.hapus = Toplevel()
        self.hapus.title("Hapus Data")
        self.hapus.geometry("580x360+120+120")

        self.list2 = Listbox(self.hapus, height=22, width=60)
        self.list2.grid(row=2, column=0, rowspan=6, columnspan=2)
        self.list2.bind('<<ListboxSelect>>',
                        self.get_selected_row)

        sb1 = Scrollbar(self.hapus)
        sb1.grid(row=2, column=2, rowspan=6)
        self.list2.config(yscrollcommand=sb1.set)
        sb1.config(command=self.list2.yview)

        tb1 = Button(self.hapus, text="Lihat Semua", command =
                     self.view_all)
        tb1.place(x=409, y=180)

        tb2 = Button(self.hapus, text="Hapus",
                     command=self.data_delete)
        tb2.place(x=495, y=180)

        tb3 = Button(self.hapus, text="Tutup",
                     command=self.hapus.destroy)
        tb3.place(x=500, y=325)

        self.nama_textx = StringVar()
        self.e1a = Entry(self.hapus, textvariable=self.nama_textx)
        self.e1a.place(x=409, y=35)

        self.nocard_textx = StringVar()
        self.e2a = Entry(self.hapus,
                         textvariable=self.nocard_textx)
        self.e2a.place(x=409, y=70)

        self.noplat_textx = StringVar()
        self.e3a = Entry(self.hapus,
                         textvariable=self.noplat_textx)
        self.e3a.place(x=409, y=105)

        self.roda_textx = StringVar()
        self.e4a = Entry(self.hapus, textvariable=self.roda_textx)
        self.e4a.place(x=409, y=140)

    def view_all(self):
        self.list2.delete(0, END)
        for row in database.view():
            self.list2.insert(END, row)

    def get_selected_row(self,event):
        try:
            index = self.list2.curselection()[0]
            self.selected_tuple = self.list2.get(index)
            self.e1a.delete(0,END)
            self.e1a.insert(END,self.selected_tuple[1])
            self.e2a.delete(0, END)
            self.e2a.insert(END,self.selected_tuple[2])
            self.e3a.delete(0, END)
            self.e3a.insert(END,self.selected_tuple[3])
            self.e4a.delete(0, END)
            self.e4a.insert(END,self.selected_tuple[4])
        except IndexError:
            pass

    def data_delete(self):
        database.delete(self.selected_tuple[0])
        self.view_all()

    def perbarui_data(self):
        self.perbarui = Toplevel()
        self.perbarui.title("Perbarui Data")
        self.perbarui.geometry("580x360+120+120")

        self.list3 = Listbox(self.perbarui, height=22, width=60)
        self.list3.grid(row=2, column=0, rowspan=6, columnspan=2)
        self.list3.bind('<<ListboxSelect>>',
                        self.get_selected_row_2)

        sb1 = Scrollbar(self.perbarui)
        sb1.grid(row=2, column=2, rowspan=6)
        self.list3.config(yscrollcommand=sb1.set)
        sb1.config(command=self.list3.yview)

        tb1 = Button(self.perbarui, text="Lihat Semua",
                     command=self.view_all_2)
        tb1.place(x=409, y=180)

        tb2 = Button(self.perbarui, text="Update",
                     command=self.update_data)
        tb2.place(x=495, y=180)

        tb3 = Button(self.perbarui, text="Tutup",
                     command=self.perbarui.destroy)
        tb3.place(x=500, y=325)

        self.nama_text_2 = StringVar()
        self.e1b = Entry(self.perbarui,
                         textvariable=self.nama_text_2)
        self.e1b.place(x=409, y=35)

        self.nocard_text_2 = StringVar()
        self.e2b = Entry(self.perbarui,
                         textvariable=self.nocard_text_2)
        self.e2b.place(x=409, y=70)

        self.noplat_text_2 = StringVar()
        self.e3b = Entry(self.perbarui,
                         textvariable=self.noplat_text_2)
        self.e3b.place(x=409, y=105)

        self.roda_text_2 = StringVar()
        self.e4b = Entry(self.perbarui,
                         textvariable=self.roda_text_2)
        self.e4b.place(x=409, y=140)
    def view_all_2(self):
        self.list3.delete(0, END)
        for row in database.view():
            self.list3.insert(END, row)

    def get_selected_row_2(self, event):
        try:
            index = self.list3.curselection()[0]
            self.selected_tuple_2 = self.list3.get(index)
            self.e1b.delete(0, END)
            self.e1b.insert(END, self.selected_tuple_2[1])
            self.e2b.delete(0, END)
            self.e2b.insert(END, self.selected_tuple_2[2])
            self.e3b.delete(0, END)
            self.e3b.insert(END, self.selected_tuple_2[3])
            self.e4b.delete(0, END)
            self.e4b.insert(END, self.selected_tuple_2[4])
        except IndexError:
            pass

    def update_data(self):
        database.update(self.selected_tuple_2[0],
                        self.nama_text_2.get(),
                        self.nocard_text_2.get(),
                        self.noplat_text_2.get(),
                        self.roda_text_2.get())
        self.view_all_2()

    def all_database(self):
        self.data = Toplevel()
        self.data.title("List Database")
        self.data.geometry("365x440+120+120")

        self.list4 = Listbox(self.data, height=22, width=60)
        self.list4.grid(row=2, column=0, rowspan=6, columnspan=2)

        sb1 = Scrollbar(self.data)
        sb1.grid(row=2, column=2, rowspan=6)
        self.list4.config(yscrollcommand=sb1.set)
        sb1.config(command=self.list4.yview)

        tb1 = Button(self.data, text="Lihat Semua",
                     command=self.view_all_3)
        tb1.place(x=141, y=384)

    def view_all_3(self):
        self.list4.delete(0, END)
        for row in database.view():
        self.list4.insert(END, row)


window = Tk()
window.geometry("300x300+120+120")
window.config(bg="#fff")
Window(window)
window.mainloop()
```