QR CODE GENERATOR

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Abdul Jawad Khan - (AP21110010131)

Srinivas Mandadapu - (AP21110010145)

Mahendra Kumar Velicheti - (AP21110010193)

Mandalaneni sesha sai - (AP2111001011182)

Under the Guidance of

(POONAM YADAV)

SRM University-AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh - 522 240

[December ,2022]

Certificate

Date: 6-Dec-22

This is to certify that the work present in this Project entitled "QR **CODE GENERATOR"** has been carried out by **Group 7 (Abdul Jawad khan, Srinivas Mandadapu, Mahendra Kumar, Mandalaneni sesha sai)** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in the **School of Engineering and Sciences**.

Supervisor

Mrs. Poonam Yadav

Acknowledgments

I would like to thank my teacher Poonam Yadav who gave me the opportunity to work on this project. I got to learn a lot from this project about the Tkinter library and pyqrcode module. I am very grateful to her for her support and guidance in completing this project. Finally, I want to thank all my teammates as well for being part of the team and for helping me complete this project.

Table of Contents

Contents

Abstract

In this project we develop QR code generator. The use of QR code-based technologies and applications has become prevalent in recent years where QR codes are accepted to be a practical and intriguing data representation / processing mechanism amongst worldwide users. The aim of this study is to design and implement a system that can generate a QR code based on the input string.

Statement of Contributions

This was the work contributed by the team members:

Abdul Jawad Khan – Coding, Research, and Report

Srinivas Mandadapu – Research and Formatting GUI

Mahendra Kumar – Debugging and Report

Mandalaneni Sesha Sai - Debugging and Report

1. Introduction

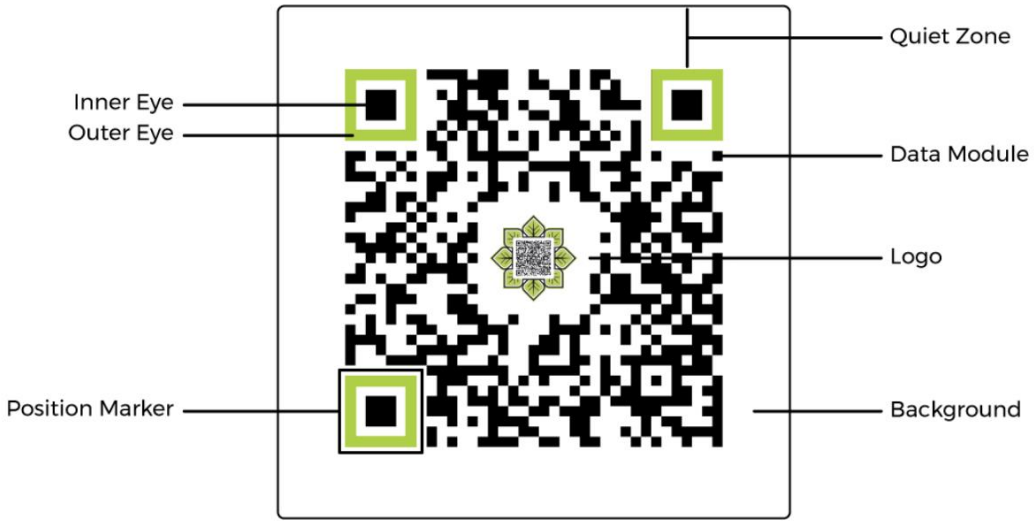
A QR (quick response) code is a box-shaped matrix form of a **2-dimensional barcode** (also called checkerboard-type barcode) that contains some meaningful data or linkage. It is frequently employed on mobile apps and websites nowadays. As compared to traditional barcodes, these data can store more information. Such code renders a certain box-type pattern across it for recording meaningful data or a large set of information in a black-and-white pattern.

QR code is the trademark for a type of matrix barcode (or two-dimensional barcode) first designed for the automotive industry in Japan. A barcode is a machine-readable optical label that contains information about the item to which it is attached. A QR code uses four standardized encoding modes (numeric, alphanumeric, byte/binary, and kanji) to efficiently store data.

The QR Code system has become popular outside the automotive industry due to its fast readability and greater storage capacity compared to standard UPC barcodes. Applications include product tracking, item identification, time tracking, document management, general marketing, and much more.

A QR code consists of black modules (square dots) arranged in a square grid on a white background, which can be read by an imaging device such as a camera. An active QR Code is one that a QR Code scanner can scan easily, prompting users to click on an external link upon scanning. The external link delivers relevant information that the user is looking for.

1.1 Parts of the QR Code



The most important parts of a QR code are:

- **Data module**: This is the standard unit of the QR code. It's typically a black square set against a white background. Though the colors and contrast can be different, black-on-white is the most optimal when creating a custom QR code. The arrangement of these black squares, or data modules, is what makes up the majority of a QR code.

- **Position marker**: There are three position markers on every QR code. Consisting of an inner and outer eye, they allow scanners and cameras to locate the data modules and the scanning direction quickly and accurately.

- **Quiet zone**: This is the blank area on all sides of the data module matrix that contains all the data modules and position markers. It allows scanners and readers to optically place where the QR code begins and ends.

2. Methodology

2.1 DESIGN:

This project has been designed majorly with the help of pyqrcode module and the Tkinter GUI library which is available in python.

- The pyqrcode module is used to create the QR code when the user gives inputas a string.
- The Tkinter library is used to create a Graphical User Interface (GUI) so the output is more interactive and easily readable by the end user.

2.1.1 Pyqrcode module:

The pyqrcode module is used to generate the QRcode in python.

The method used from the pyqrcode module:

create(): This method is used to create the QR code of any input string. The syntaxfor using this method is as follows:
pyqrcode.create(*content*, *error='H'*, *version=None*, *mode=None*, *encoding=None*).
However, when creating a QR code only the content to be encoded is required, all the other properties of the code will be guessed based on the contents given.This function will return a QR code object.

E.g. qr=pyqrcode.create("input string from the user")

2.1.2 Tkinter LIBRARY:

Tkinter is the standard GUI library for Python.

Widgets in Tkinter

In this, there are various widgets available with which the user can either interact/view. Following are a few Tkinter widgets we used to design this project:

Label: A Label is a Tkinter Widget class, which is used to display text or an image. Thelabel is a widget that the user just views but does not interact with.

Button: The Button widget is a standard Tkinter widget, which is used for various kinds of buttons. A button is a widget that is designed for the user to interact with, i.e. if the button is pressed by mouse click some action might be started.

Entry: Entry widgets are the basic widgets of Tkinter used to get input, i.e. text strings, from the user. This widget allows the user to enter a single line of text. Ifthe user enters a string, which is longer than the available display space of the widget, the content will be scrolled. This means that the string cannot be seen inits entirety.

LabelFrame: LabelFrame is a simple container widget. It creates a mini frame inside the main frame. It can contain other buttons/labels in it.

Attributes of widgets in Tkinter:

bg: The normal background color is displayed behind the label and indicator.
fg: This option specifies the color of the text (foreground color).
padx: Extra space added to the left and right of the text within the widget.
pady: Extra space was added above and below the text within the widget.
font: If you are displaying text in this label (with the text or text variable option), the font option specifies the style, size, and other characteristics (i.e. bold, italic, etc. of the font, and in this style, the text will be displayed.

text: To display one or more lines of text in a widget

command: This is an attribute that is used in the button widget. This is assigned tothe function which is to be called when the button is clicked.

Layout managers in Tkinter:

Tkinter has three Layout Managers that use geometric methods to position widgets in an application frame namely the pack (), place(), and grid() methods. The Layout managers we used in this project are:

pack (): It organizes widgets in horizontal and vertical boxes that are limited to left, right, top, and bottom positions. Each box is offset and relative to the other.

place (): It places widgets in a two-dimensional grid using x and y absolute coordinates.

grid (): It locates widgets in a two-dimensional grid using row and columnabsolute coordinates.

Some other Tkinter methods we used in this project:

config(): This method is used for performing an overwriting over a label widget

2.2 IMPLEMENTATION

2.2.1 Code:

```
      from tkinter import *
from tkinter import messagebox
import pyqrcode

ws = Tk()

def generate_QR():
    if len(user_input.get()) != 0:
        global qr, img
        qr = pyqrcode.create(user_input.get())
        img = BitmapImage(data = qr.xbm(scale = 5))
    else:
        messagebox.showwarning('warning', 'All Fields are Required!')
    try:
        display_code()
    except:
        pass

def display_code():
    displayimage.config(image = img,bg='white')
    output.config(text = "SUCCESSFULLY GENERATED THE QR CODE OF YOUR INPUT ",
font=("Terminal",15),bg='white', fg='black')

#*************************************************************************************
************************************(end of backend)
#GUI STARTS

def team_info():
    f3 = LabelFrame(ws,bg =
'#00B9F1',highlightthickness=0,borderwidth=0,padx=5,pady=25)
    f3.pack()
    f3.place(x=30,y=600)

    lheading = Label(f3,text = "DONE BY: ",font=("bauhaus 93",20),bg = "#002E6E",
fg='white',padx=130)
    l2 = Label(f3,text="Jawad Khan",font=("Arial rounded MT
bold",20),highlightthickness=0,borderwidth=0,padx=83,bg='white')
    l3 = Label(f3,text="Srinivas M",font=("Arial rounded MT
bold",20),highlightthickness=0,borderwidth=0,padx=95,bg='white')
    l4 = Label(f3,text="Mahendra Kumar",font=("Arial rounded MT
bold",20),highlightthickness=0,borderwidth=0,padx=50,bg='white')
    l5 = Label(f3,text="Sesha Sai",font=("Arial rounded MT
bold",20),highlightthickness=0,borderwidth=0,padx=98,bg='white')
    lheading.grid(row=0,column=0,pady=10)
    l2.grid(row=2,column=0,padx=30,pady=10)
```

```
    l3.grid(row=3,column=0,padx=30,pady=10)
    l4.grid(row=4,column=0,padx=10,pady=10)
    l5.grid(row=5,column=0,padx=10,pady=10)

#*************************************************************************************
****************************************

lmain=Label(ws,text= "QR CODE GENERATOR",font=("bauhaus
93",30),bg='#00B9F1',fg='white',padx=675)
lmain.pack()
lmain.place(x=0,y=20)

#*************************************************************************************
*********************************(end of main heading)

f1 = LabelFrame(ws,bg = '#00B9F1',highlightthickness=0,borderwidth=0,padx=5,pady=50)
f1.pack()
f1.place(x=30,y=100)

l1 = Label(f1,text = "ENTER ANY STRING TO GENERATE THE QR CODE ",font=("bauhaus
93",20),bg = "#002E6E", fg='white',padx=50)
l1.grid(row=0,column=0,pady=10)

user_input = StringVar()

e = Entry(f1,textvariable = user_input,font=("Terminal",20),width=35)
e.grid(row=1,column=0,pady=50)

b1 = Button(f1,text="GENERATE QR",font=("bauhaus
93",20),highlightthickness=0,borderwidth=0,command=generate_QR,padx=50,bg='white')
b2 = Button(f1,text="EXIT",font=("bauhaus
93",20),highlightthickness=0,borderwidth=0,command=ws.destroy,padx=105,bg='white')
b3 = Button(f1,text="TEAM INFO",font=("bauhaus
93",20),highlightthickness=0,borderwidth=0,command=team_info,padx=65,bg='white')
b1.grid(row=2,column=0,padx=10,pady=10)
b2.grid(row=3,column=0,padx=10,pady=10)
b3.grid(row=4,column=0,padx=10,pady=10)

#*************************************************************************************
*********************************(end of frame 1)

f2=LabelFrame(ws,highlightthickness=0,borderwidth=0,bg='white')
f2.pack()
f2.place(x=900,y=100)

displayimage = Label(f2, bg= "#002E6E")
displayimage.grid(row=0,column=0)
```

```
output = Label(f2,text = "",bg = "#002E6E")
output.grid(row=1,column=0)

#*********************************************************************************
*********************************(end of frame 2)

ws.attributes('-fullscreen',True)
ws.config(bg="#002E6E")
ws.mainloop()
```

2.2.2 Explanation of code:

- First, we will import tkinter and all its sub-modules using the *.
- Then we will also import messagebox and pyqrcode. Tkinter module helps in creating GUI windows and apps where users can interact with events.
- We will generate object ws by assigning it with the Tk() constructor.
- Then we create a user-defined function named **generate_QR()**

  def generate_QR():

  This is a user defined function where all our QR code logic will reside. The pyqrcode.create() generates the QR code as it fetches the string through **user_input.get()** method from the text box. Also, this function will get executed when the user_input.get() is not equal to 0. Once the qr code gets generated using the pyqrcode.create(), we have to use the **BitmapImage()** and pass the data = qr.xbm() along with a scale size (here 5) that will generate a Bitmap image of 5x5. If the user_input.get() is equals 0, then the control goes

to the else block where **messagebox.showwarning()** shows a warning to fill the data in that text box. Then within the try block we call the display_code().

- **def display_code():**

  This user-defined function displays the QR code and shows the message

  "SUCCESSFULLY GENERATED QR CODE OF YOUR INPUT". It is a function that

  shows the image details as well as the status info that we can see at thebottom of that application.

- **def team_info():**

  This user-defined function is to display the names of the team members by whom this project is done. It has an object f3 of LabelFrame widget. This object f3 is created inside the main frame ws. We set the required attributes ofthis widget. Inside this frame, we have different Label objects namely lheading,l2,l3,l4, and l5 which are used to show the names of the team members.

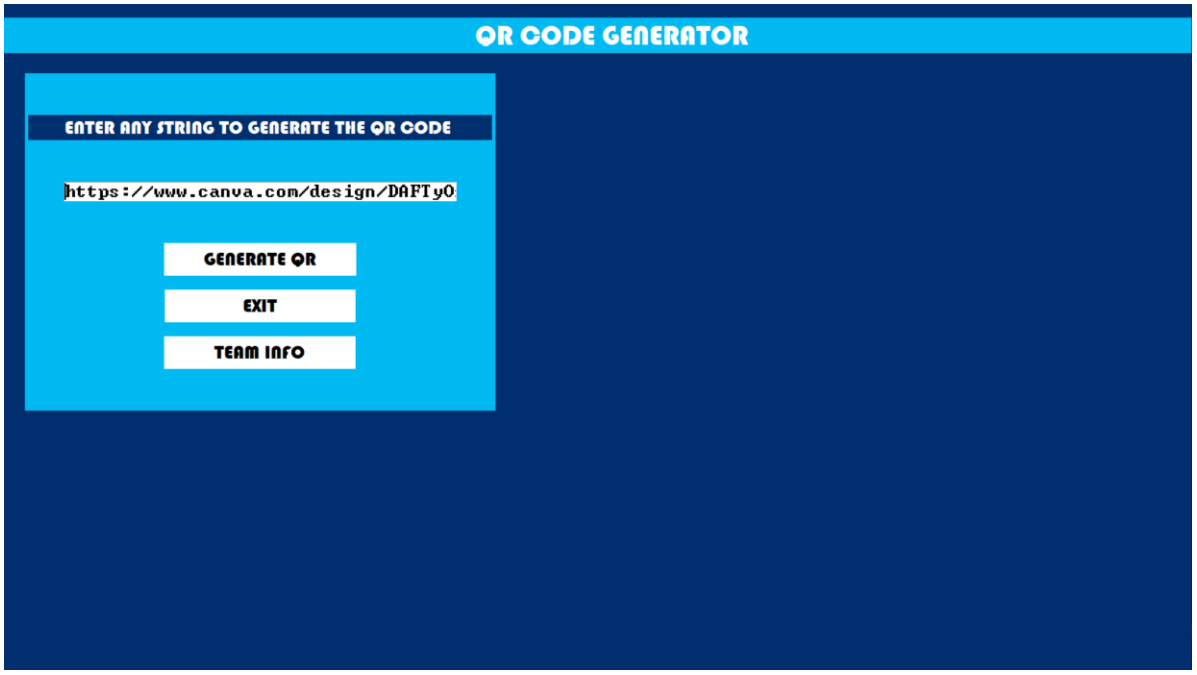
- Throughout, we have to use the **Label(), Entry(), and Button()** constructors of the Tkinter module to generate the labels, text box, and buttons. We will then store these Constructor initialization in different objects namely, lmain, l1 ,f1 ,f2 ,e ,b1 ,b2 ,b3.

- We have also used 'command' in the required button, which when clicked by the user control goes to that specific function.
  Eg: Button b1 has the text "GENERATE QR". When user click on this button the control must go to generate_QR() function. This is done using command=generate_QR

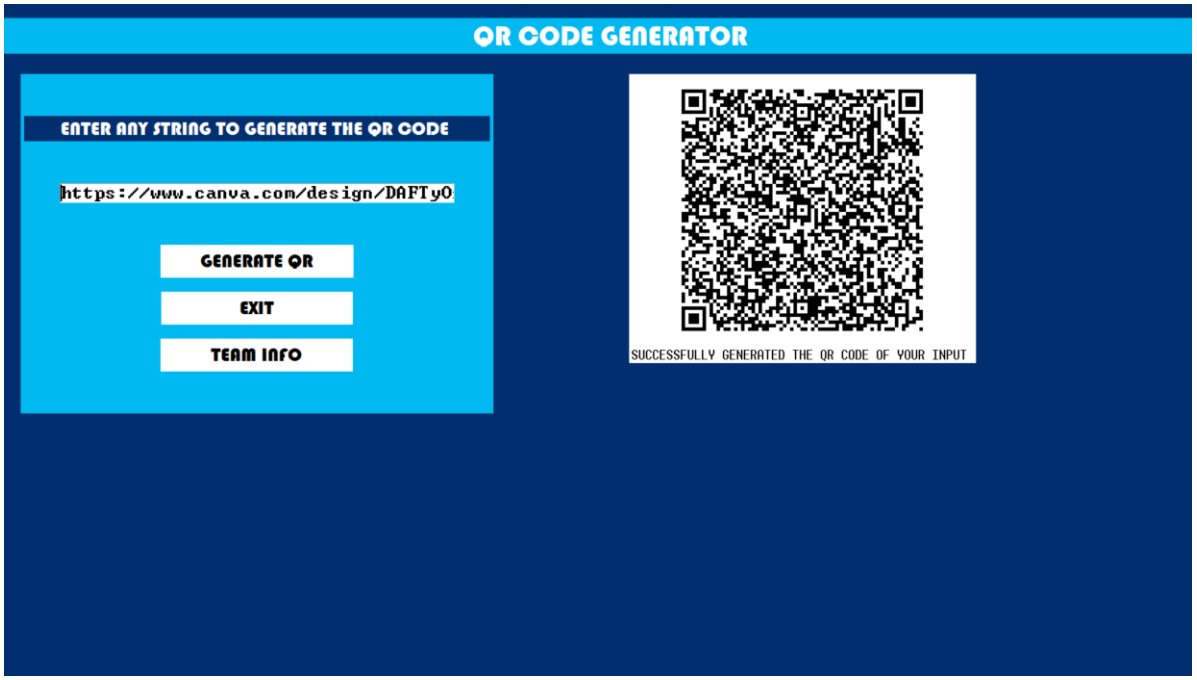
- Lastly, the mainloop() is used, which is an infinite loop implemented for running the application, waiting for an event to occur, and processing theevent as long as the window does not get closed by the user manually.

3. Results

User giving the input string (site link) in entry box

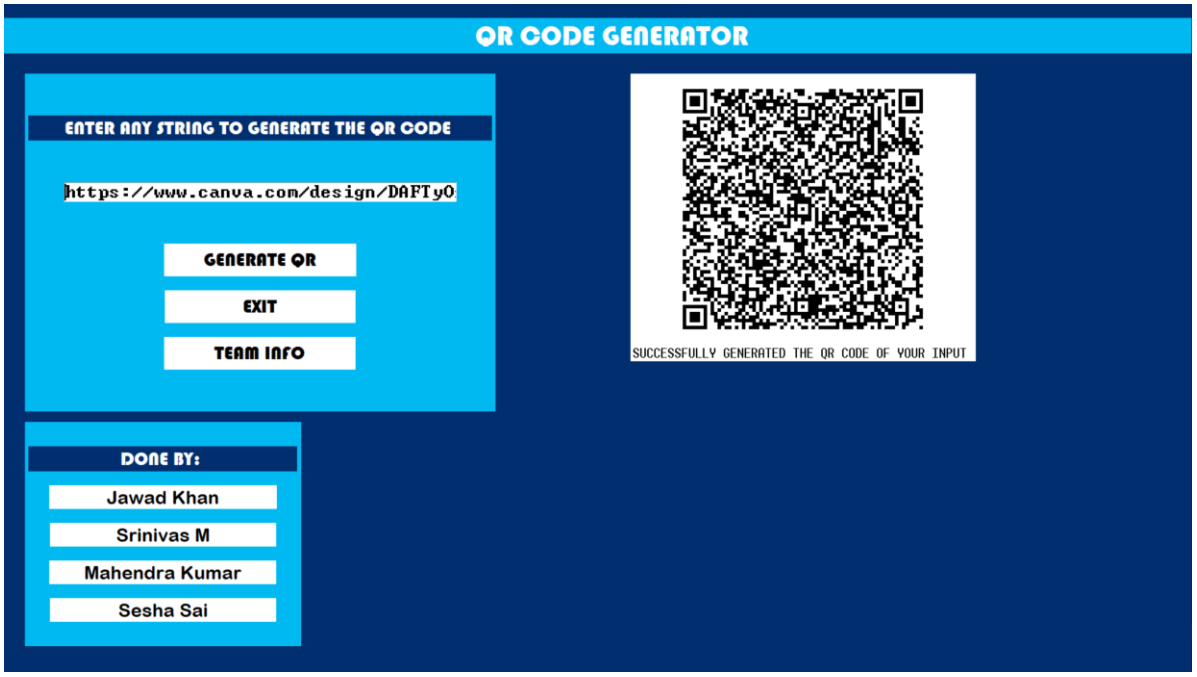


Clicking the 'GENERATE QR' button



After pasting the site link, when the user clicks on "GENERATE QR" button, a unique QR Code of the site is generated.

Clicking the “TEAM INFO” button



After generating the QR Code if the user clicks on "TEAM INFO" button, a frame displaying all the team members appears. (This step isn't absolutely necessary).

Clicking the “EXIT” button:

After this, if the user the user clicks "EXIT" button, the QR Code Generator interface gets closed.

4. Conclusion

In the process of making this project we have understood about Python Libraries and how to implement them. This project has many real-life applications like generating a unique QR code for a business website, etc.