MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION (MUMBAI)

Academic Year
2024 - 2025

MICRO-PROJECT REPORT

"QRyCODEx : QR Code Generator using Flask "

This Micro-Project developed under the,

Program Name: Computer Engineering ***Semester:6th***

Course Title: PWP ***Course Code:22616***

Submitted by,

Name of Group Member : Shaikh M.M. **Enroll. No: 221205******

Under the guidance of,

(Mr. Chougule V.V.)

(DTE CODE : D-6474 / MSBTE CODE-1205)
SHREE VATAVRUKSHA SWAMI MAHARAJ DEVASTHAN's
KAI. KALYANRAO (BALASAHEB) INGALE POLYTECHNIC COLLEGE, AKKALKOT.
NBA Accredited Programs
Gat No. 604/2, Near Bhakt Niwas, Ganagapur Road, Akkalkot-413216. Dist. Solapur. (Maharashtra)
Ph.: (02181) 221321, E-mail : swamipolytechnic@gmail.com • Website : www.swamipolytechnic.org

(DTE CODE : D-6474 / MSBTE CODE-1205)

SHREE VATAVRUKSHA SWAMI MAHARAJ DEVASTHAN's

KAI. KALYANRAO (BALASAHEB) INGALE POLYTECHNIC COLLEGE, AKKALKOT.

NBA Accredited Programs

Gat No. 604/2, Near Bhakt Niwas, Ganagapur Road, Akkalkot-413216. Dist. Solapur. (Maharashtra)

Ph.: (02181) 221321, E-mail : swamipolytechnic@gmail.com • Website : www.swamipolytechnic.org

Certificate

This is to certify that the following group of students submitted Micro-Project Report entitled "QRyCODEx : QR Code Generator using Flask " of 6th Semester of Diploma in Computer Engineering has completed the Micro-Project satisfactorily in course PWP (course code:22616) for the academic Year 2024 to 2025 as prescribed in the curriculum.

Submitted by

Name of Group Member : Shaikh M.M. **Enroll. No: 221205******

…… ……

Micro-Project Guide **H.O.D.**

Departmet

Seal

S.V.S.M.D's

Kai. Kalyanrao (Balasaheb) Ingale Polytechnic, Akkalkot

(Affiliated to MSBTE MUMBAI)

2024 - 2025

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION (MUMBAI)

Academic Year
2024 - 2025

MICRO-PROJECT REPORT

"QRyCODEx : QR Code Generator using Flask "

This Micro-Project developed under the,

Program Name: Computer Engineering ***Semester:6th***

Course Title: PWP ***Course Code:22616***

Submitted by,

Name of Group Member : Shaikh M.M. **Enroll. No: 221205******

Under the guidance of,

(Mr. Chougule V.V.)

(DTE CODE : D-6474 / MSBTE CODE-1205)
SHREE VATAVRUKSHA SWAMI MAHARAJ DEVASTHAN's
KAI. KALYANRAO (BALASAHEB) INGALE POLYTECHNIC COLLEGE, AKKALKOT.
NBA Accredited Programs
Gat No. 604/2, Near Bhakt Niwas, Ganagapur Road, Akkalkot-413216. Dist. Solapur. (Maharashtra)
Ph.: (02181) 221321, E-mail : swamipolytechnic@gmail.com • Website : www.swamipolytechnic.org

Annexure-I A

PART A – PLAN

"QRyCODEx : QR Code Generator using Flask"

1.0 Brief Introduction:

QR codes (Quick Response codes) are widely used to store information, such as links, texts, or other data. They can be easily scanned with a mobile phone camera to quickly access information without manually typing it. This project aims to develop a simple and efficient web-based tool where users can input data, and the application will generate and display a corresponding QR code.

The Flask framework is used to create this web application. Flask is a lightweight, easy-to-use web framework that is perfect for small projects like this one. We will use Flask to handle user input and display the generated QR code. To generate the QR code, the project will leverage the Python qrcode library, which allows easy creation of QR codes from text or URLs.

Technologies Used:

- **Flask**: A web framework for Python used to handle HTTP requests and render HTML pages.
- **qrcode Library**: A Python library used to generate QR codes from user input.
- **HTML/CSS**: Used to create the user interface of the web application.
- **Python**: The programming language used to develop the back-end functionality of the app.

2.0 Aim of the Micro-Project:

- **Create a Web Application:** Develop a web-based tool that allows users to input data (text, URLs) and generate corresponding QR codes.
- **Implement Flask for the Back-End:** Use Flask to handle user requests, manage data, and display the generated QR code.
- **Generate QR Codes with the qrcode Library:** Utilize the Python qrcode library to create QR codes based on the user's input.
- **Design a Simple User Interface:** Build an easy-to-use front-end where users can input their data, view the generated QR code, and have the option to download it.
- **Integrate Front-End and Back-End Technologies:** Combine Flask (back-end) with HTML/CSS (front-end) to create an interactive and functional web application.
- **Learn Web Development Basics:** Enhance understanding of Flask, Python libraries, and web development principles by creating a practical tool for QR code generation.

3.0 Action Plan:

Sr. No.	Details of activity	Planned Start date	Planned Finish date	Name of Responsible Team Members.
1	Topic Search	03-02-2025	10-02-2025	All team member
2	Discuss with our guide	10-02-2025	17-02-2025	All team member
3	Search through reference book	17-02-2025	24-02-2025	All team member
4	Assign topics	24-02-2025	02-03-2025	All team member
5	Divide task	02-03-2025	09-03-2025	All team member
6	Start to implement project	09-03-2025	16-03-2025	All team member
7	Dubbling error	16-03-2025	23-03-2025	All team member
8	Represent demo	23-03-2025	30-03-2025	All team member

Resources Required:

Sr. No.	Name of Resource / Material	Specification	Quantity	Remarks
1	Computer system	Computer (i3-i5 preferable)	1	-
2	RAM	2 GB	1	-
3	Operating system	Windows	1	-
4	Software	MS Word	1	-

-

4.0Any other:

Conclusion :

This project demonstrates the use of Flask for building simple web applications and showcases how Python can interact with external libraries like qrcode to create practical tools. By developing this QR Code Generator, the project highlights the combination of web development and utility creation in a practical, easy-to-understand format.

We request you to please accept our Micro-Project Proposal and we assure you of our sincere performance. Hoping for your co-operation & guidance with us.

Sr. No.	Name of Student	Roll Number	Course Code	Student Sign.
1	Shaikh M.M.	242318	22616	

Micro-Project Guide **H.O.D**

PART B

"QRyCODEx : QR Code Generator using Flask"

1.0 Brief Introduction:

QR codes (Quick Response codes) are widely used to store information, such as links, texts, or other data. They can be easily scanned with a mobile phone camera to quickly access information without manually typing it. This project aims to develop a simple and efficient web-based tool where users can input data, and the application will generate and display a corresponding QR code.

The Flask framework is used to create this web application. Flask is a lightweight, easy-to-use web framework that is perfect for small projects like this one. We will use Flask to handle user input and display the generated QR code. To generate the QR code, the project will leverage the Python qrcode library, which allows easy creation of QR codes from text or URLs.

- **How the Application Works:**
  - **User Input:** The user visits the webpage and enters the data (like a URL or a piece of text) they want to encode into a QR code.
  - **Generate QR Code:** After the user submits the data, the Flask app processes the input, uses the qrcode library to generate the QR code, and displays it back to the user.
  - **Download or Scan:** The user can either download the generated QR code or scan it directly with their mobile phone.

- **Technologies Used:**
  - **Flask:** A web framework for Python used to handle HTTP requests and render HTML pages.
  - **qrcode Library:** A Python library used to generate QR codes from user input.
  - **HTML/CSS:** Used to create the user interface of the web application.
  - **Python:** The programming language used to develop the back-end functionality of the app.

- **Features of the Application:**
  - **Input form for entering data (URLs, text).**
  - **QR code generation on the server side.**
  - **Display of generated QR code as an image on the webpage.**
  - **Option to download the QR code image.**

- **Imports and Setup**
  - **Flask:** Used to create the web server and handle routes (URLs).
  - **qrcode:** A library to generate QR codes.
  - **BytesIO**: Allows handling images in memory instead of writing them to disk.
  - **os and tempfile:** Used for working with the file system, like creating temporary files and directories.

  The Flask app is initialized with app = Flask(__name__). A temporary directory is created for storing QR codes using tempfile.mkdtemp().

- **Routes**
  - **/about:** Renders an "About" page (About.html).
  - **/home:** Renders a "Home" page (index.html).
  - **/ (Index):** The main route where a user can generate a QR code. If the user sends a POST request (e.g., submitting a form):
    - The app will read the URL, fill color, and background color from the form.
    - It generates a QR code with those parameters using the generate_qr() function.
    - It stores the generated QR code in a temporary file and provides two URLs: one for viewing the image and another for downloading it.
- **QR Code Generation**
  - generate_qr(url, fill_color, back_color): This function creates a QR code using the qrcode library with the user-provided URL and colors (fill and background). It returns the generated image.
- **Image Viewing and Downloading**
  - /generate_qr_code: This route sends the generated QR code image as a response so it can be displayed on the webpage.
  - /download_qr_code: This route sends the QR code image as a downloadable file when the user clicks the download link.
- **Report Download**
  - /download_report: This route allows the user to download a PDF report (Report.pdf) stored in the static folder of the app
- **Running the App**
  - The app runs in debug mode, which makes it easier to develop by showing errors and auto-reloading when changes are made.
- **Key Flow:**
  - User visits the home page (/home).
  - User submits a URL and color choices in a form.
  - The app generates a QR code with those settings.
  - The user can view or download the generated QR code.

- **Example Usage:**
  - Generate QR code: You provide a URL and choose colors for the QR code, and then you can either see it or download it.
  - Download Report: A static PDF report can be downloaded using the /download_report route.
- **Security/Practical Considerations:**
  - The code uses temporary directories to store QR codes, which are deleted automatically after the server stops.
  - It provides a way for users to download the QR code and report, but doesn't include any advanced security checks (like validating the URL input).
- CODE:

```
from flask import Flask, render_template, request, send_file, url_for
import qrcode
from io import BytesIO
import os
import tempfile
app = Flask(__name__)
TEMP_DIR = tempfile.mkdtemp()
@app.route('/about')
def about():
    return render_template('About.html')
@app.route('/home')
def home():
    return render_template('index.html')
@app.route('/', methods=['GET', 'POST'])
def index():
    qr_code_url = None  # Will store the URL of the generated QR code
    download_url = None  # URL to download the generated QR code
    if request.method == 'POST':
        url = request.form.get('url')
        fill_color = request.form.get('fill_color')
        back_color = request.form.get('back_color')
        img = generate_qr(url, fill_color, back_color)
        img_io = BytesIO()
        img.save(img_io, 'PNG')
        img_io.seek(0)
        temp_filename = os.path.join(TEMP_DIR, "qr_code.png")
        with open(temp_filename, 'wb') as f:
            f.write(img_io.read())
        qr_code_url = url_for('generate_qr_code')  # Use the route for viewing the image
        download_url = url_for('download_qr_code')  # Use the route for downloading the image
    return render_template('index.html', qr_code_url=qr_code_url, download_url=download_url)
```

```
@app.route('/generate_qr_code')
def generate_qr_code():
    if os.path.exists(os.path.join(TEMP_DIR, "qr_code.png")):
        return send_file(os.path.join(TEMP_DIR, "qr_code.png"), mimetype='image/png')
    return "QR Code not found", 404
@app.route('/download_qr_code')
def download_qr_code():
    temp_filename = os.path.join(TEMP_DIR, "qr_code.png")
    if os.path.exists(temp_filename):
        # Provide the image for download
        return send_file(temp_filename, as_attachment=True, download_name='qr_code.png',
mimetype='image/png')
    return "QR Code not found", 404
def generate_qr(url, fill_color, back_color):
    # Create QRCode instance
    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_H,
        box_size=10,
        border=4
    )
    qr.add_data(url)
    qr.make(fit=True)
    img = qr.make_image(fill_color=fill_color, back_color=back_color)
    return img
@app.route('/download_report')
def download_report():
    # Define the path to the report PDF file in the static folder
    report_path = os.path.join(app.root_path, 'static', 'Report.pdf')
    if os.path.exists(report_path):
        return send_file(report_path, as_attachment=True, download_name='Report.pdf',
mimetype='application/pdf')
    return "Report not found", 404
if __name__ == "__main__":
    app.run(debug=True)
```

2.0 Aim of the Micro-Project:

- **Create a Web Application:** Develop a web-based tool that allows users to input data (text, URLs) and generate corresponding QR codes.
- **Implement Flask for the Back-End:** Use Flask to handle user requests, manage data, and display the generated QR code.
- **Generate QR Codes with the qrcode Library:** Utilize the Python qrcode library to create QR codes based on the user's input.
- **Design a Simple User Interface:** Build an easy-to-use front-end where users can input their data, view the generated QR code, and have the option to download it.
- **Integrate Front-End and Back-End Technologies:** Combine Flask (back-end) with HTML/CSS (front-end) to create an interactive and functional web application.
- **Learn Web Development Basics:** Enhance understanding of Flask, Python libraries, and web development principles by creating a practical tool for QR code generation.

3.0 Course Outcomes Integrated:

d) Develop functions for given problem.

e) Design classes for given problem.

f) Handle exceptions**.**

4.0 Actual Procedure Followed:

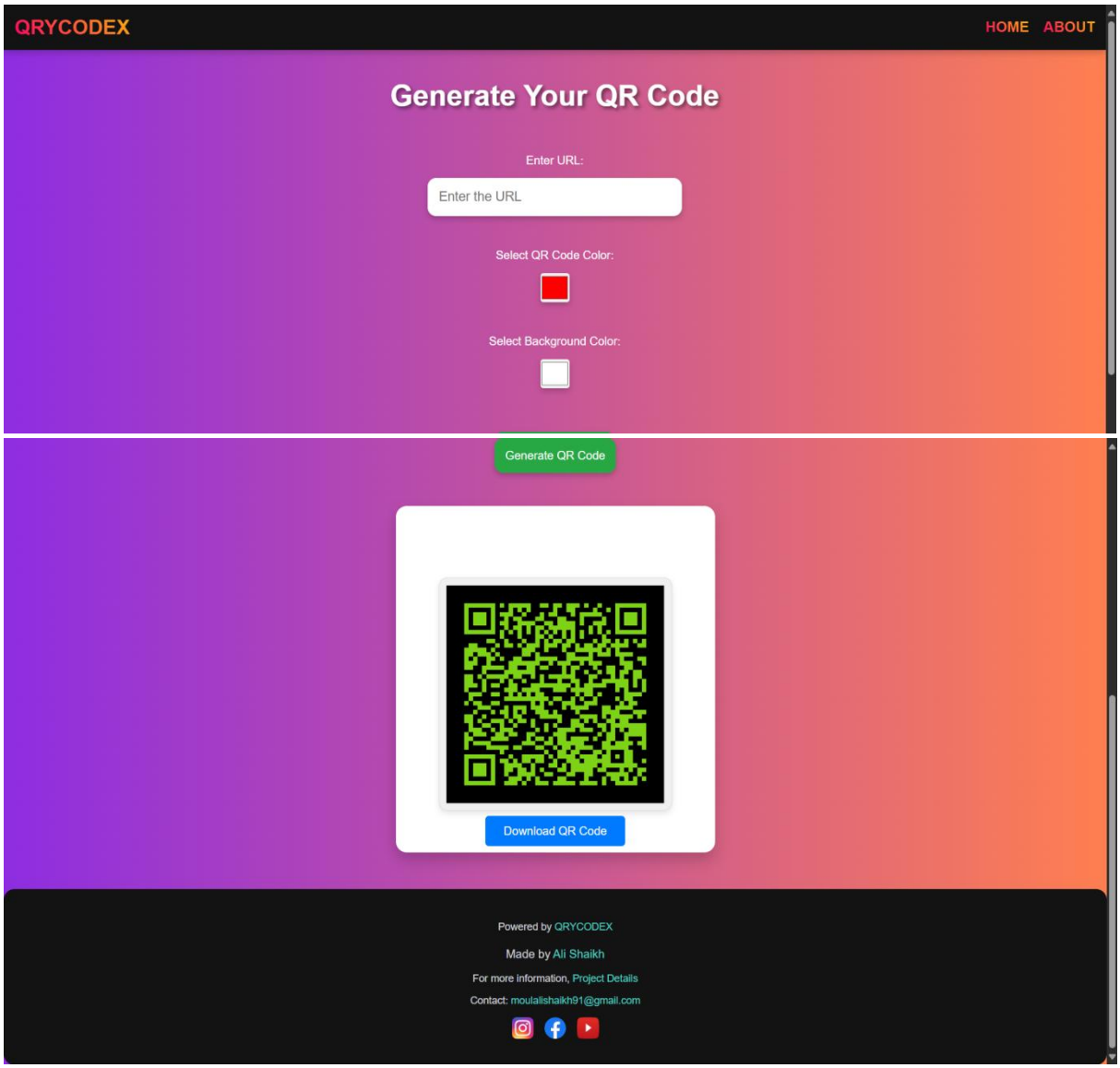
- **Topic Search:** We chose the topic of digital evidence after exploring different ideas.
- **Discuss with Guide:** We talked to our guide to get feedback and refine the project.
- **Search Through Reference Books:** We read books and online resources to gather information on digital evidence.
- **Assign Topics:** We divided the project into smaller parts, with each person responsible for a specific section.
- **Divide Task:** Tasks like research, writing, and analysis were shared among team members.
- **Start Implementation:** We wrote the sections, shared our work, and made sure everything fit together.
- **Dubbling Error:** We checked for repeated information and removed any duplicates.
- **Represent Demo:** We created a final presentation to explain our findings with examples.

5.0 Actual Resource Used:

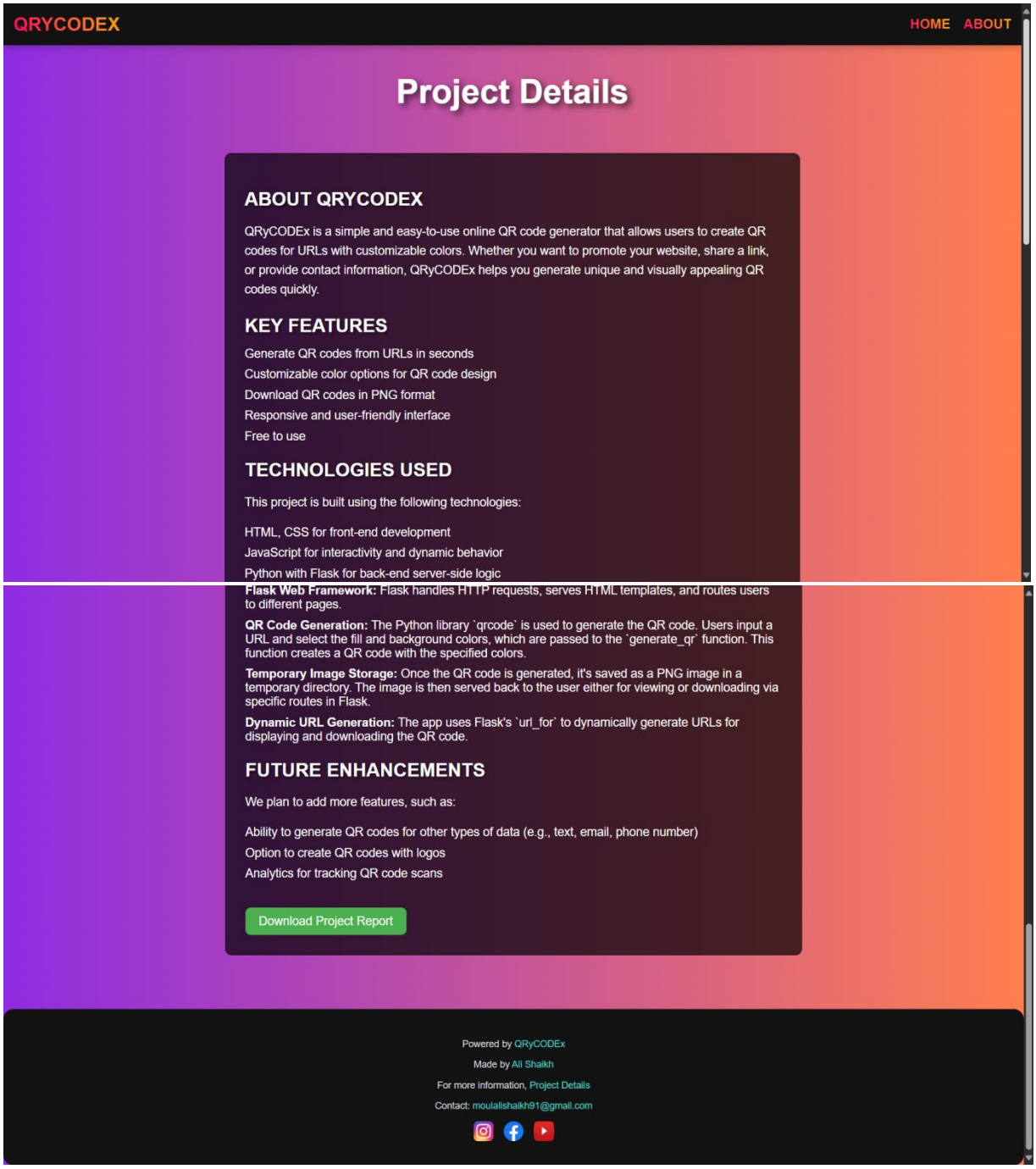
Sr. No.	Name of Resource / Material	Specification	Quantity	Remarks
1	Computer system	Computer (i3-i5 preferable)	1	-
2	RAM	2 GB	1	-
3	Operating system	Windows	1	-
4	Software	VS Code	1	-

6.0 Outputs of the Micro-Projects:

Index.html



About.html


QRYCODEX
HOME ABOUT
Project Details
ABOUT QRYCODEX
QRyCODEx is a simple and easy-to-use online QR code generator that allows users to create QR codes for URLs with customizable colors. Whether you want to promote your website, share a link, or provide contact information, QRyCODEx helps you generate unique and visually appealing QR codes quickly.
KEY FEATURES
Generate QR codes from URLs in seconds
Customizable color options for QR code design
Download QR codes in PNG format
Responsive and user-friendly interface
Free to use
TECHNOLOGIES USED
This project is built using the following technologies:
HTML, CSS for front-end development
JavaScript for interactivity and dynamic behavior
Python with Flask for back-end server-side logic
Flask Web Framework: Flask handles HTTP requests, serves HTML templates, and routes users to different pages.
QR Code Generation: The Python library `qrcode` is used to generate the QR code. Users input a URL and select the fill and background colors, which are passed to the `generate_qr` function. This function creates a QR code with the specified colors.
Temporary Image Storage: Once the QR code is generated, it's saved as a PNG image in a temporary directory. The image is then served back to the user either for viewing or downloading via specific routes in Flask.
Dynamic URL Generation: The app uses Flask's `url_for` to dynamically generate URLs for displaying and downloading the QR code.
FUTURE ENHANCEMENTS
We plan to add more features, such as:
Ability to generate QR codes for other types of data (e.g., text, email, phone number)
Option to create QR codes with logos
Analytics for tracking QR code scans
Download Project Report
Powered by QRyCODEx
Made by Ali Shaikh
For more information, Project Details
Contact: moulalishaikh91@gmail.com

- **File Structure :**

```
QRyCODEx/
│
├── app.py              # Main Python file that runs the Flask application
│
├── static/             # Folder for static files like images, CSS, JS, and PDFs
│   └── report.pdf      # The report PDF file
│
└── templates/          # Folder for HTML template files
    ├── index.html      # Home page template
    └── About.html      # About page template
```

- **Explanation of the structure:**

  QRyCODEx/: The root folder of your Flask app containing the main Python script and other folders.

  - **app.py:** The main Flask application file that contains the routes and logic for generating QR codes, serving pages, and downloading reports.
  - **static/:** Folder for storing static files (files that don't change, like images, stylesheets, and downloadable files).
    - **report.pdf:** A sample PDF report that can be downloaded by the user.
  - **templates/:** Folder for storing HTML templates used to render pages in the web app.
    - **index.html:** The template for the home page, where users can input data to generate QR codes.
    - **About.html:** The template for the "About" page, providing information about the app.

7.0 Skill Developed / learning out of this Micro –Project:

- **Flask Web Development:** Learn to create web applications, handle routes, and process user requests.
- **QR Code Generation:** Understand how to generate and customize QR codes using the qrcode library.
- **Handling User Input:** Learn how to manage form data and adjust app behavior based on user input.
- **File and Directory Management:** Work with temporary directories and manage files using os and tempfile.
- **Making Content Downloadable:** Learn to serve downloadable files like images and PDFs in a web app.
- **Frontend Integration:** Use HTML templates and forms to create interactive web pages.
- **Debugging:** Gain experience debugging web applications in development mode.

8.0 Any other:

Conclusion :

This project creates a simple web app that lets users generate QR codes. Using Flask and the qrcode library, users can input data like URLs or text, and the app will create a QR code for them. They can then see the QR code on the page or download it.

The app uses Flask to handle the user's input and create the QR code in real time. It also saves the QR code temporarily, so users can access or download it whenever needed.

This project helped develop skills in web development, like handling forms, working with files, and generating dynamic content. Flask is a great tool for building small web apps, and this project showed how to use it effectively.