Ticketing System

This project generates, prints, and scans tickets with QR codes. It uses Python, Flask, ReportLab, and PyPDF2.

1. Prerequisites

- Python 3.8+ installed on Windows (64-bit).
- Git Bash or PowerShell for OpenSSL (if using HTTPS).
- Wi-Fi network set to **Private** profile on Windows.

2. Clone and Install

1. Clone the repo into your folder:

   ```
   git clone <repo-url> ticketing_system
   cd ticketing_system
   ```

2. Create and activate a virtual env:

   ```
   python -m venv .venv
   .venv\Scripts\activate
   ```

3. Install dependencies:

   ```
   pip install -r requirements.txt
   ```

3. Generate Tickets and QR Codes

1. Edit `config.py` and adjust the number of tickets `N_TICKET` according to your needs.

2. Run script:

   ```
   python generate_ticket_ids_and_qrcodes.py
   ```

3. Check outputs:

   - `tickets.db` holds the predefined number of IDs.
   - `qr_codes/` contains PNGs plus `scan_page.png`.
   - `ticket_urls.csv` maps IDs to URLs.

4. Create Ticket PDFs

1. Place your designer PDF as `ticket_template.pdf` into the project's root folder.

2. If needed, adjust the dimensions of the ticket in `config.py`.

3. Run generation script:

   ```
   python generate_tickets_pdf.py
   ```

4. Check `tickets/` folder for individual ticket PDFs.

5. Run the Flask Server

1. Create SSL cert and key (ensure `"/CN=192.168.1.18"` matches your `HOST` in config.py):

   ```
   export MSYS_NO_PATHCONV=1
   openssl req -x509 -newkey rsa:2048 -nodes \
     -keyout key.pem -out cert.pem -days 365 \
     -subj "/CN=192.168.1.18"
   ```

2. Start server with HTTPS:

   ```
   python app.py
   ```

   The console should show:

   ```
   * Running on https://0.0.0.0:8000/
   ```

3. If using `flask run`, add flags:

   ```
   flask run --host=0.0.0.0 --port=8000 --cert=cert.pem --key=key.pem
   ```

6. Scanner Webpage

- URL: `https://192.168.1.18:8000/scan`
- Page uses live camera and ZXing JS to decode QR.
- Message appears large at center.
- 3 seconds delay between scans.

7. User Instructions

1. Connect to the **Private** Wi-Fi (not a guest network).

2. Open your browser (preferably Safari) and go to the scan URL.
3. Accept the self-signed certificate (click **Erweitert → Trotzdem fortfahren**).
4. Allow all required permissions, including the page to use the camera.
5. Do not close the browser tab while scanning.
6. If you accidentally close it, clear site data and cache:
    - In browser settings: **Cookies & Websitedaten**, **Cache → Löschen**.
7. Re-open the scan URL and allow camera.

8. Troubleshooting

- **Server unreachable**: check LAN IP, firewall, port-forwarding, AP isolation. Try clear the site and data cache of your browser as described in 7.6.
- **Camera not available**: ensure HTTPS, clear old cache, disable ad-blocker.
- **First scan works, then fails**: use the continuous-scan page; do not rely on built-in camera app handoff.
- **Internal Server Error**: Ask Daniel for help. Something might be wrong with the code.