Terri Rodriguez-Hong

Developer's Guide

This application is Python/Flask app that creates a webpage for users to read QR codes and stores associated data in a server side SQLite3 database.

The User flow:

1. User launches a Safari or Chrome browser in a mobile phone or tablet PC and go to the website: https://flaxenink.pythonanywhere.com
   (critical that you use https not http or webcam won't work, no picture taking)
2. Once you are on the homepage of the application, you can select from 3 different application views (Closet, Add Item, and Scan QR Code) from the navigation at the bottom of the app. Each of these views are routes in the flask_app.py file at the root of the application directory on your webserver. This flask_app.py file is the only python file used in the application. No other python is used outside this file.
3. Html pages are created from html templates in the templates folder and are referenced in the routes. Data is retrieved and stored in a sqlite database from python and passed into html templates as variables.
4. To view all saved QR Code items in a list, select **Closet**. You can then an item description to see the item detail or delete an item clicking on the associate trash can. When the trashcan icon is pressed, javascript calls a location change to "/delete_item/<qrid>" route with the value of the qrid number. Python then uses this passed in qrid value to remove the record in the database with that qrid. The arrow at the end of each item also launches the item description when clicked. A separate route in flash_app.py is targeted through a Javascript location change. This location change is where a qrid number is passed to the "/display_qrcode/<qrid>" route. This route function block receives the qrid value as a parameter and then queries the QRCLOTHES table in sqlite to find the associated record for that item.
5. To **Add an Item**, click on the "add an item" button on the bottom of the screen. A javascript change location calls the "add_new_item" route and the route then finds the last known qrid number and passes it to the html "additem.html" template. This value is incremented to the next new qrid and a qrcode is generated by javascript in the browser. The user has the option to enter notes for that item as well as click on the "click to take picture" button to launch an overlay for taking a picture from the webcam. In the overlay, you can take as many versions of the picture as you want by clicking take picture over and over. The X button closes the overlay without saving your picture. The "save" button saves the picture data in the item form. The "Save Item" button submits the form that includes all of the invisible item data with a POST method to a "save_data" route in python. All new item data is collected and added as a new item record in the sqlite database with the new qrid.
6. To **Scan a QR Code**, click the button at the bottom right footer of the screen. This button calls a javascript location change to route "readqrcode". The route displays the qrcode.html template file in the browser. This view uses an external javascript qrcode library to scan qrcode from live webcam images. Once a qrcode is recognized in javascript, a javascript location change to '/get_data_with_id/<qrid>' where qrid is the

code extracted from the recognized qrcode image. This new route then uses the qrid to query the database for the record associated with that qrid and launches a new webpage based on the display_qrcode.html template (with passed in item variables).

INSTALLATION (how to run locally):

1. Download the app from my repository:
   git clone https://github.com/trodriguezhong/QRCodeClothing_HCI_584
2. Make sure python3.9 is installed with "python –version" and "Pip install flask" in the terminal / command line to install flask.
3. Make sure sqlite is installed: sqlite3 –version. Should be version 3 or higher
4. Make sure you have python and the following packages installed:
   a. ssl
   b. sqlite3
   c. logging
5. Run flask from command line: FLASK_APP=flask_app_local.py flask
   (for localhost:5000)

NOTE:

The final project was uploaded to: https://flaxenink.pythonanywhere.com/
Users can go to this page to use the application.
Only a single user (smiles@yahoo.com) and no user authentication is provided at this time.

TODOS and possible future features

1. Add user authentication
2. Multiple user support
3. Portrait and landscape views (only Portrait supported now)
4. Error checking for new QRCodes and ability to re-use qrids. Needs to show more elegant html error template page other than crashing the app.
5. Error checking and default values for images so that the front end can evaluate if an image was provided and show an alternate image for no image data returned.
6. Need ability to edit notes and update images in the closet view. Maybe add pencil icon next to trash can to edit.
7. Convert app to PWA to allow installing app icon on desktop and run in chromeless browser.
8. Move css in to external shared css files. I learned the hard way that changes in too many places instead of in a shared file is not good. E.g. nav.css, styles.css etc…
9. Fix issues with viewing app on many different device screens and types