Dokumentation

GitHub Benutzername: mawgit

Dokumentation: Inhalt

1) Einleitung und Projektidee
2) Definitionen, Hintergrundinformationen, Begründungen
3) Projektplan/Projektaufbau
   - Beschreibung der einzelnen Elemente und Funktionen
   - Beschreibung der Werkzeuge, Hilfsmittel, etc.
   (wie ist das Programm aufgebaut und warum wurde es so aufgebaut)
4) Anleitung zur Ausführung/Benutzung
5) Reflexion
   - Wurde das Ziel erreicht? Wenn nicht, wieso?
   - Was ist schiefgelaufen? Wieso? Lösungen?
   - Fazit (wobei hier nichts Neues erläutert, sondern das Projekt noch einmal kurz zusammengefasst und die eigene Erfahrung/das Ergebnis bewertet werden soll)

---

1) Einleitung und Projektidee

Meine Projektidee war eine Flask (Web-) APP zu erstellen, die GoogleMaps Karten beinhaltet um Parkscheinautomaten und Busparkplätze in Köln anzeigen zu können.

Umgesetzt werden sollte das Projekt dann mit Python, Flask, GoogleMaps und Javascript.

Als erstes habe ich auf eine offene Datenquelle zugegriffen um die Koordinaten der georeferenzierten Parkscheinautomaten und Busparkplätze in Köln auf GoogleMaps herauszufinden. Diese speicherte ich dann im Json-Format ab und verarbeitete sie mit Python. Ich erstellte eine statische Karte wo alle Parkscheinautomaten / Busparkplätze im Umkreis in unterschiedlichen Farben angezeigt werden.
Diese Karte soll man nur sehen können wenn man sich bei Flask eingeloggt hat.
wenn man sich bei Flask eingeloggt hat.

2) Definitionen, Hintergrundinformationen, Begründungen

Ich habe erstmal versucht eine Flask Anwendung zu erstellen, zu erweitern und dann erst bei Github reinzustellen um nicht jeden einzelnen Buchstaben committen zu müssen. Das habe ich erst beim wichtigeren Teil gemacht.

Meinen Projektfortschritt habe ich auf einem USB Stick gespeichert und nach zu langen Internetverbindungsproblemen dann von einem anderen Computer aus das Projekt bei Github upgeloaded. Dabei habe ich vergessen vorher "git config --global user.name " einzugeben, deshalb wurde vom System stattdessen Windows als User.name eingegeben. Änderung war nicht mehr möglich, doch nachdem ich dann git config gemacht habe, hat es bei den nächsten commits wieder mit meinem richtigen Git user.name geklappt.

3) Projektplan/Projektaufbau

- Beschreibung der einzelnen Elemente und Funktionen

Ich habe versucht mein Projekt sowohl unter Python 2.7.12, als auch unter Python 3.5.2 lauffähig zu machen. Zum Teil gelang es mir, doch es wurde schwierig, dazu schreibe ich später genaueres.

Verwendete Hilfsmittel und Werkzeuge sind:
Python 3.5.2
Python 2.7.12
Flask 0.11.1
Google MAPS API

Es gibt zwei MAPS, eine für Parkscheinautomaten und eine für Busparkplätze. Zur besseren Übersicht wurden die beiden getrennt.

In den beiden MAPS kann man Mit den PLUS- und MINUS-Tasten, die sich rechts unten in der MAP befinden, heran oder heraus-zoomen.
Links oben auf der MAP kann man sich entscheiden ob man die Karte oder Satelliten Ansicht wählen möchte, die markierten Parkscheinautomaten und Busparkplätze bleiben dabei bestehen und werden weiterhin angezeigt.

Als Erweiterung habe ich noch die zusätzlichen Informationen der georeferenzierten Parkscheinautomaten und Busparkplätze hinzugefügt, so dass man bei einem Klick auf eine Markierung von einem Parkscheinautomat z.B. den Aufstellort, Bezirk_Gebiet, den Abschnitt_Von, Abschnitt_Bis, Stellplaetze, Gebuehr, Hoechstparkdauer, Gebuehrenzeit aufgelistet bekommt; und bei Busparkplätzen: BEZEICHNUN, GEBUEHREN, NUTZUNGSART sehen kann.

Die templates generieren die HTML Seiten die in Flask angezeigt werden, darunter der Login und die Index Seite.

In der parkautomaten.py ist das Wurzelverzeichnis, es wird eine Datenbank angelegt und das Passwort gehasht.

4) Anleitung zur Ausführung/Benutzung

Mein Projekt wurde von mir getestet und ist funktionsfähig mit Python 2.7
Flask==0.11.1
Flask-Babel==0.11.1
Flask-Login==0.3.2
Flask-WTF==0.12
Jinja2==2.8

Um es zum laufen zu bringen muss man zunächst eine vernünftige Python Installation haben. Danach sollte man folgende Module installieren:

pip install Flask
pip install Flask-Login
pip install Flask-Babel
pip install Flask-WTF

Dann ruft man die Python Konsole auf und gibt folgende Kommandos ein:
>>> from database import init_db
>>> init_db()
Dadurch wird die Datenbank initialisiert.

Dann gibt man in die Konsole den Aufruf ein:
>>> python parkautomaten.py

Zum starten der Flask-APP muss man nun folgendes in den Browser (z.B. Firefox oder Chrome) eingeben:
http://127.0.0.1:5000/
http://localhost:5000/

5) Reflexion

Das Ziel wurde zum Teil erreicht. Unter Python 2.7 läuft die Flask Anwendung, man kann sie starten und sich mit den Zugangsdaten:
User: admin, Password: dedefault
anmelden und bekommt die GoogleMaps Karten mit Parkscheinautomaten und Busparkplätzen angezeigt, aber unter Python 3.5 hat es leider noch nicht funktioniert.

Eine Überlegung zur Erweiterung war noch ein Bewertungs oder Favorisierungs-System für die Parkscheinautomaten und Busparkplätze in Flask zu erstellen, doch wie soll das umgesetzt werden? Die klickbaren Markierungen in Javascript auf der MAP sind schwierig in Flask zu übermitteln und als Favorit zu speichern, dann müsste die MAP für jeden User neu erstellt werden und anders aussehen, denn jeder User kann andere Favoriten oder Bewertungen für die Markierten Parkscheinautomaten abgeben. Markierungen müssten dann zusammengefügt werden und Favoriten nur für einen selbst sichtbar sein. Ohne hilfreiche Tipps zur Umsetzung wäre das zu schwierig.

Was ist schiefgelaufen?

Ich musste mich durch viele Fehler in meiner Flask APP durchkämpfen und es kamen leider immer mehr neue Fehlermeldungen hinzu.
Die meisten davon haben mich viel Zeit und Nerven gekostet, diese zu bewältigen.
Um den dadurch entstandenen zusätzlichen Arbeitsumfang und Frust zu verstehen, schreibe ich mal ein paar der Fehlermeldungen auf:

OperationalError: (OperationalError) no such table: users u'SELECT users.id AS users_id, users.username AS users_username, users.password AS users_password, users.active AS users_active \nFROM users \nWHERE users.username = ?\n LIMIT ? OFFSET ?' (u'admin', 1, 0)

OperationalError: (OperationalError) unable to open database file None None

sqlalchemy.exc.OperationalError: (sqlite3.OperationalError) no such table: users [SQL: 'SELECT users.id AS users_id, users.username AS users_username, users.password AS users_password, users.active AS users_active \nFROM users \nWHERE users.username = ?\n LIMIT ? OFFSET ?'] [parameters: ('admin', 1, 0)]

NameError: name 'unicode' is not defined

NameError: name 'xrange' is not defined

TypeError: ord() expected string of length 1, but int found

sqlalchemy.exc.InvalidRequestError: No transaction is begun.

binascii.Error: Incorrect padding

etc.

Fazit:

Das Projekt hat viel mehr Zeit und Arbeitsaufwand beansprucht als erwartet.
Mit Python zu programmieren ist zwar interessant und macht Spaß, aber die Probleme mit Flask sind leider frustrierend gewesen.

Das Troubleshooting war nicht immer erfolgreich. Die meisten Fehler kamen beim Versuch die Flask Anwendung auch unter Python 3.5.2 zum laufen zu bekommen.
Ich habe lange versucht die „binascii.Error: Incorrect padding“ Fehlermeldung (die dabei nur unter Python 3.5 kommt) zu lösen, doch nach vielen vergeblichen Versuchen gelang es mir nicht. Die Fehleranalyse der Flask-Error-Meldungen und auch Recherchen mit Google und Stackoverflow ergaben leider keinen Erfolg beim Lösen des Problems.

Quellenangaben:

Die Quellen der Daten zu Parkscheinautomaten und Busparkplätzen sind:

Literaturverzeichnis: (2016): „Parkscheinautomaten in Köln“. Offene Daten Köln. Abgerufen am 23.08.2016 von http://www.offenedaten-koeln.de/dataset/parkscheinautomaten-koeln.

Literaturverzeichnis: (2016): „Busparkplätze in Köln“. Offene Daten Köln. Abgerufen am 23.08.2016 von http://www.offenedaten-koeln.de/dataset/busparkpl%C3%A4tze-k%C3%B6ln.