Artifacts List: Release 1

Sebastian Sherry n9161350

# Admin.html and admin.js:

Admin.html and admin.js come together to form the admin portal for the project. The html file is very basic, as most of the page is generated through javascript. The page is the basic skeleton table and a button leading to a page to add violations and heath issues.

The javascript uses Jquery’s getJson method to access the database using specially made urls that access the database. This data is then used to populate the tables. Each table is generated when selected from a radio button in the html.

# db.py and dbTest.py:

Db.py implements the database connector for the project. Each table has their own insert and update function, while the retrieve and delete functions are universal for all tables. The connector is implemented as class for easy use in lookup.py. The file also contains a main function used to input the initial test values into the database. This function is usually run to reset the database for examples or when new tables/values are added.

DbTest.py is a script written to test the initial database implementation to prove all the connector functions work. Theses tests do not prove that the database will work in a production environment, as they lack the required structure to but unit tests. This script fills more of a proof of concept role, with better tests to be written later in the development cycle.

# Lookup.py:

Lookup.py is the bridge between views and the database connector. It contains various methods to perform back end data manipulation and retrieval. The included functions include (but are not limited to): Login credential checking, retrieving parking permits and all violations, and adding permits and violations to the database. All functions requiring database access have their own database object created and closed at the end of the function to prevent issues with multiple users accessing the database at once.

# Profile.html:

Profile.html is a simple page that displays all relevant user data for any user on the database (besides admin). Using functions from lookup.py and flask’s jinja2 templating, the page is quite dynamic. For example, if the user doesn’t have a parking permit, the table displaying the permit information is replaced with text stating they don’t have a permit. Similarly, if a user doesn’t have a parking violation, the table isn’t shown.

When the user does have a violation, they can find all relevant data on this page. Each violation has a button which will take the user to payment form when implemented.

# Views.py:

Views.py is the main file used to run the project. When executed by the server, it displays and handles the website. This file defines all the pages on the site, which ones require the user to be logged in or an admin, and also passes data to and from lookup.py functions to pages where required. The file also contains two helper functions which help format data and initialize cookies.

Sessions are extensively used throughout the site. These sessions help keep track of user information such as email and if they have admin privileges. This data is useful for retrieving relevant data and access control to the admin portal.