Technical Documentation

Main Folder

There are many files required to carry out the functions of this web site. Each of these files can be opened with any text editing software (such as Notepad on Windows), although it is strongly encouraged to use a source code editor (such as Notepad++ or Netbeans) in order to view the markup of the file easier. In order to keep everything sorted, all the files were separated based on the purpose they serve. The main folder contains the pages that the user will actually see and interact with. The files found here are written in both HTML and PHP and are each explained in better detail below:

index.php

This is the home page of the web site and it is used as the basis for the design of the pages that follow it. Like any HTML document, the code begins by referencing a couple essential files. These files include the 'style.css' file for the design and the Javascript code to run the slideshow seen across the middle of the page. From this page the user can log in, register, go to the 'View Garage' page, or go to the 'Reservations' page. If the user tries to access something outside these boundaries, it will re-direct them back to the home page as they require the user to log in first. If the user tries to log in without invalid log on credentials, the site will re-direct them back to the home page. Lastly, the user can go any of the other three routes available to them without logging in as they don't require any specific account information.

myAccount.php

When a user logs in or creates an account, they will be re-directed to this page. This will be the page the user will be able to maintain the information relating to their account. Now that the user has logged in, it is essential to create and keep track of a session for this user. There are three forms on this page that allow the user to update/add information to their account. Each of these three forms call *'include/changeSettings.php'* to carry out the proper change based on the value of change. To change the password, change is 1, for adding a credit card, change is 2, and to add a vehicle, change is 3. Another key thing to note here is that the menu bar has changed. When a page is loaded, the system will verify whether a proper session has been established and provide an option to log out if it is.

register.php

The register page allows a new user to create an account with very little information to start. All the fields seen on this page are required and will not proceed without them. If a user tries to enter an email that already exists in the database, enters two different email addresses, or leaves one field blank, the appropriate error message will be returned. This is all done via a form that calls 'include/newAccount.php' that verifies this and finally adds the user if all is valid.

reservation.php

This is where the user will be able to place their reservation. There are dropdown menus to allow the user to provide an exact start and end point. There are two errors that can be generated off this page. The first one occurs when the user selects a start point that comes before the end point. The other obvious error will occur when there are too many conflicts for the times and dates requested and the reservation cannot be processed. For now these are the only options offered on this page, but future implementations will include allowing the user to select which car they wish to use for the reservation as well as the option to place a guaranteed reservation via a credit card on file. On a successful request, the system will return the reservation ID and the spot assigned to it.

viewGarage.php

Whether the user has an account or not they can access this page. The purpose of this page is to give the user an idea of the current capacity of the garage. It pulls the current time and generates a table that shows whether a spot is vacant or not. It does this by calling the function *'createGarageMap()'* located in *'include/extraFunctions.php'* every time the page is loaded.

viewReservations.php

Just like the View Garage page, this page does not require a user to log in to view. It allows the user to get an idea of the reservations currently placed in the system. By default it will show the next twenty reservations based off the current time. The user can input a specific timeframe using the dropdown menus, which are identical to the ones seen on the reservation page. In order to protect user privacy, the only information displayed are the start points, end points, and spot numbers assigned to a reservation.

CSS Folder

In that same main folder, there is a sub-folder named *'css.'* CSS stands for cascading style sheets and greatly simplifies designing each individual page as HTML code can get pretty lengthy. The key file in this folder is *'style.css'* as it contains all the classes used to design the elements of the website. The HTML code found in the files of the main folder all reference this file as they all are based off the same design template.

JS Folder

Moving along, the next sub-folder is entitled 'js' and it contains the necessary code for the sliding window on the home page of the web site. Its main purpose is just to run a slideshow where it will pull the necessary pictures from the *'css/images'* folder. The code here is written in Javascript and was taken from a free web site template found online.

Include Folder

All the files in the main folder do not carry out any actual functions for the web site. Instead they gather user information and reference files via forms. The files they reference are actually located in this sub-folder. These files include:

authorize.php

When a user attempts to log in or out, this file is called to verify whether the action is valid. When a user logs out, their session is no longer valid and they are re-directed back to the home page. When a user logs in, the credentials are checked in the database to ensure it is a valid existing user.

changeSettings.php

Like mentioned above, this file is essential to making the actual account changes. If all is successful, the user will be re-directed back to their account page with no errors. Otherwise, depending on the error, the appropriate message will be displayed.

config.inc.php

This file is essential to grant access to a certain database. It goes into greater detail in the '4_data_collection' folder where each entry is explained based on how the user plans to simulate the site.

db.php

This file is just a long list of database calls. The functions found here can be used in one of two ways, depending on whether information needs to be returned or not. If the purpose of the database call is to update the information, the function will just take the input information and carry out the necessary task. If any information needs to be returned, every entry pull from the database is stored as an index of an array. This array is then returned to the file that called it to carry on.

extraFunctions.php

Almost every page on this site will reference this file. To begin, the menu bar located at the top of each page is rendered based on whether the user is logged in or not through the function *'renderMenu.'* The following function *'createGarageMap()'* will pull the real time and display a table of all the spots in the parking garage. If a spot is vacant the background color for the cell will be green and if the spot is occupied, it will be red. The system knows the status of the spot based on pulling all the reservations that conflict with the current time. The next two functions, *'generateRealTimeResTable'* and *'generateCustomResTable,'* display current reservations on the system seen on the Reservations option on the menu. The real time table uses the current time to pull the upcoming reservations and the custom table takes the user's start and end points and produces a table of reservations based on that. The last three functions are just functions to ease the dropdown menus seen through the site. They vary on whether they require past or future dates.

fetchReservations.php

When a user tries to customize their view on the Reservations page, their start and end points will be validated on this page. Once the system has confirmed that the two points are valid, it will return them in the URL for display in *'extraFunctions.php.'*

newAccount.php

This file pulls the information the user entered on the registration page and validates it before making any additions to the database. It ensures that all fields have been entered, the two email entries match, and that the email address requested is unique. Once this is all confirmed, it references the function *'registerUser'* in *'db.php'* to carry out the registration process and takes the user to their account page.

newReservation.php

Much like *'fetchReservations.php,'* this file takes the user reservation request and will validate the start and end points. Once this is done, it will check to make sure there is enough space in the parking garage to make this reservation by pulling the database reservations that conflict with it. If the reservation is valid, the system will go ahead and assign a parking spot to it, based on availability, through the database. Lastly, it will insert the reservation and return the reservation ID to the user for their records.

verify.php

This file hasn't been fully implanted yet and may not be. Originally this was used to check whether a session was valid or not. This is now done at the beginning of each page that would require user information and eliminates the purpose of the file. For now this will just be left alone in case any future changes require it.