Student ID: 18016286

Login details for default user in car company website: Phone number = **1111111111** Password = **cane**

Website path: http://localhost/18016286/index.php

Number of words: 1250

Task	**Comments**
1. Car company webpage	Page shows categories of cars using ajax, has links to login and register page. When user is logged in order history and place order buttons are present too.
2. Submit the SQL files to generate the tables in the car company	SQL files are working and have been placed into the "sql" folder in the project directory.
3. Login page	Existing users can login with the correct combination of phone number and password. Live validation achieved via React.
4. Registration page	New users can be registered, provided their name as well as phone number are unique. Live validation achieved via React.
5. Security aspects (preventing attacks) of the Registration and Login page	Prepared statements have been used to prevent SQL injection attacks, passwords are stored hashed in the database so that a breach won't compromise the user's password, htmlspecialchars has been used to mitigate XSS attacks.
6. Search engine optimization	All main pages have a title tag, a meta description, keywords, and an author. Language and character set are also specified.
7. User Registration validation using React framework	Could be better, sharing state between components should be implemented in order to patch a bug that lets the user through. The server side validation still prevents us from getting the data into the db.
8. Data management	a) Customer registration table: customers that sign up are inserted into the "userer" table b) Customer ordered table: orders placed are inserted into the "carorder" table
9. Any other information (remarks)	Not much, but I would've liked to understand/use React in a better way.

Working Procedure Report for 18016286’s Car Company Website

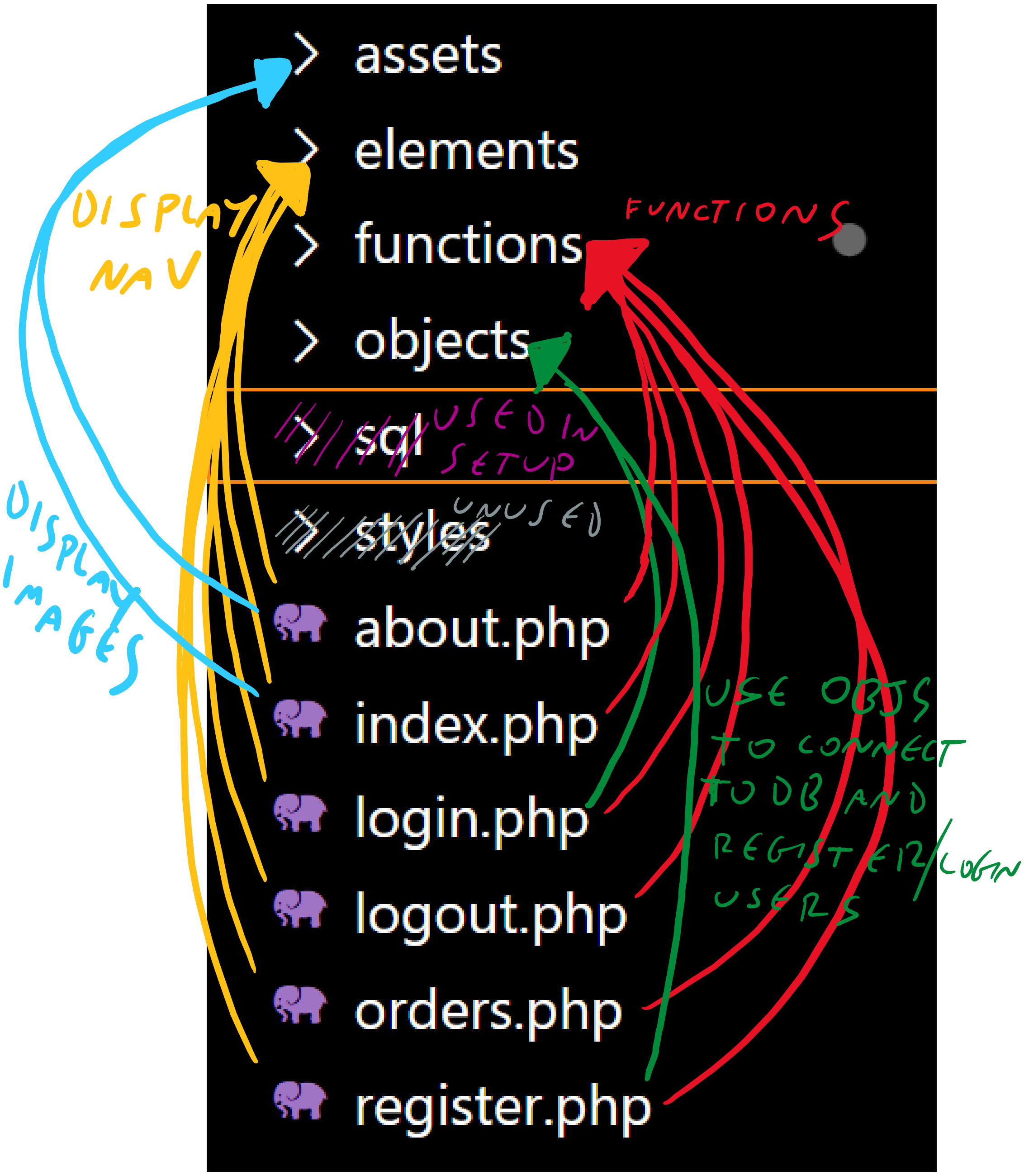
Introduction

The root directory of the project contains the main pages:

- Index.php: the homepage, the view allowing us to list different brands of cars as well as order them if we are logged in
- Register.php: the registration page, applies live and server-side validation and lets us register new users. Once registered we are asked if we want to proceed to the login page
- Login.php: the login page, here we can log in using phone and password (validation applied again), after successfully logging in we are redirected to the homepage
- Logout.php: we get to this page by clicking the logout button and we then get asked if we want to go to the homepage or login page (in case we want to login with a new user)
- Orders.php: the page displaying past orders. We only populate the page with the orders placed by the current logged in user
- About.php: credits and sources for the images

The root directory also contains a number of folders:

- Assets: folder containing the images used for the cars
- Elements: folder containing UI elements (only the navbar so far)
- Functions: folder containing PHP files doing set functions (like register_user.php, order_car.php, logout_user.php and similar)
- Objects: folder containing PHP files representing objects used in the webapp (database and user objects)
- Sql: folder containing the .sql files needed to create tables in the database and insert the 6 cars and 1 registered user
- Styles: folder that ended up not being used at all, it should’ve incorporated all of the styling files regarding the project but very little styling actually happened



assets
elements
functions
objects
sql
styles
about.php
index.php
login.php
logout.php
orders.php
register.php

Figure 1: image displaying rough relationships between the project's files and folders

Developed Architecture

The starting webpage is index.php where the user can select which brand of cars to view through a dropdown menu (using ajax). The user can browse cars along with their information but cannot see any "Order this Car" buttons or "Order History" tab as long as he isn't logged in.

The navbar remains visible through all of the website and contains the essential navigation links.

From index.php (or any other page with navigation) the user can navigate to the "About This" tab in order to check out the image sources and licences.
The user can also click on the "Register" button to sign up a new user, the username field will need to be only letters while the phone number field will need to be exactly 10 digits (0 to 9 in value).
The email will also need to be valid (something@something.something) and the password should be only made up of letters.
A number of messages will guide the user when inputting data into these fields, once everything looks good the user can press on "Register" to register the new user (implying there were no validation issues).

After having registered an account, the user is prompted to log in by a message and button. If the user follows the link (or clicks on “Login” in the upper right corner, or “Already Registered”) he will be brought to the login page.

From the login.php page the user can log in using a previously registered (or default) combination of phone number and password details and then pressing on the “Login” button. The same constraints as the register.php page apply here too. If the credentials are correct the user will be redirected to index.php where he will find each model of car having an “Order This Car” button on the side of its price, along with a new “Order History” tab added to the left side of the “About This” tab. The user can also notice that the previous “Login”/”Register” button combination in the navbar will have changed to a single “Logout” button, with a “Logged in as $username” message on its left. This will help the user keep track of the app’s state as well as reach more contextually useful controls.

At this point the user can either place an order by pressing a button or press on the “Order History” tab. Both actions will bring him to the orders.php page where he will be able to see a list of past orders and details however, clicking the “Order This Car” button will add this latest order to the list before displaying it. Orders display (just like cars display) is managed with ajax, and a given user is only able to view orders placed by himself in the list. Users are not limited to a set amount of orders and can repeatedly order the same car multiple times.

From here the user can inspect his past orders, force refresh the list via a button or navigate to any part of the website he wants to interact with again. Clicking on the “Logout” button will log the user out and bring him to a page offering the option to log in again or go to the index page (and anything else contained in the navbar).

Suggested Improvements and Conclusions

A general improvement of React’s live validation is needed. Users would benefit from being able to insert numbers or special characters in the password field, and React’s live validation shouldn’t have the bug allowing a user to submit a name with numbers by ignoring the warnings and typing letters after. This could be achieved by sharing the state between the form components.

Security could be improved depending on the requirements, and a bug that should be absolutely fixed is the email field not being correctly filled with the user’s email when placing new orders.

Images could also be resized or rearranged in a way where they don’t take up almost half of the screen, or the behaviour should be split between desktop and mobile since phones benefit from large elements.