Roll

Wei Zhou

Weijay@live.ca

ROLL-OUT cAR sHARE

Database Management System

Specifications and Functionalities of automated car rentals of the future

## Code & Scripts:

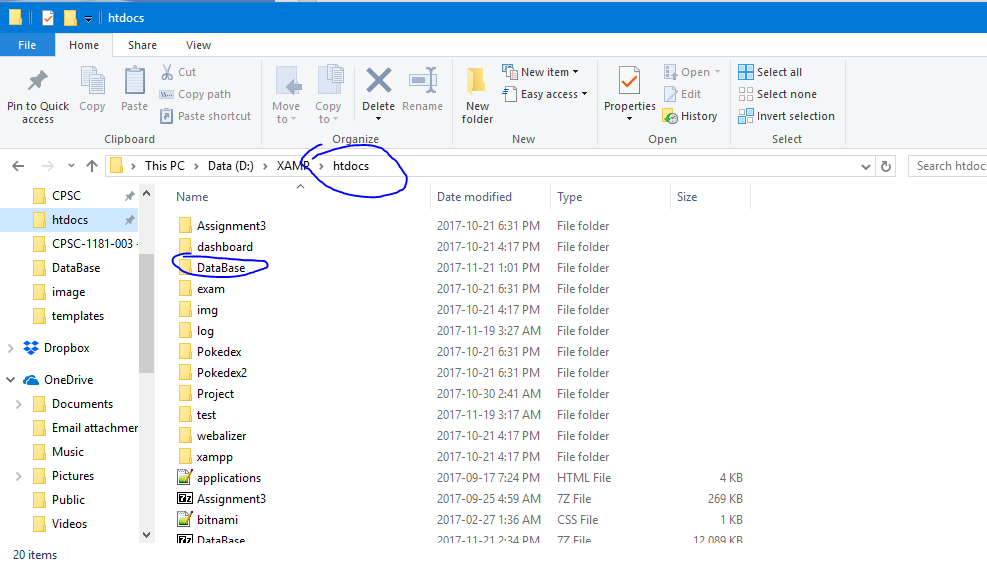
All of the necessary code has been provided in a Zip file with this report.

## Instructions for running codes:

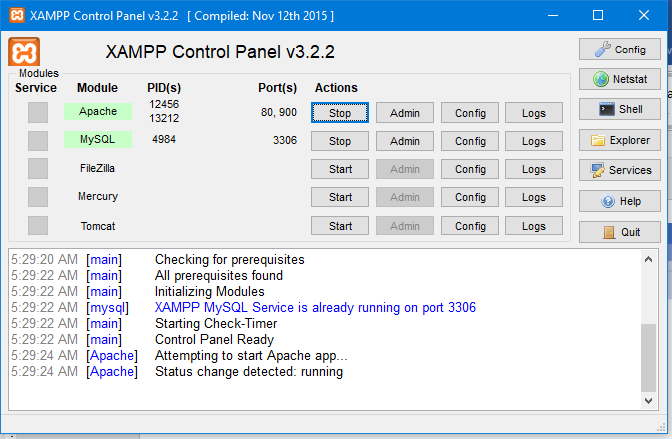
To get the code up and running, you will need to have a phpmyadmin and an apache server. For this project, I used XAMPP to emulate the environment. Here are the steps to use if you do use XAMPP:

Step 1. Install XAMPP on PC. https://www.apachefriends.org/index.html.

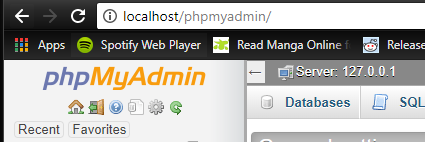
Step 2. Copy database folder included in this zip into the htdocs folder.



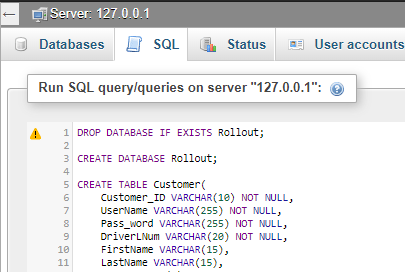
Step 3. Turn on XAMPP and use it to enable Apache and SQL.



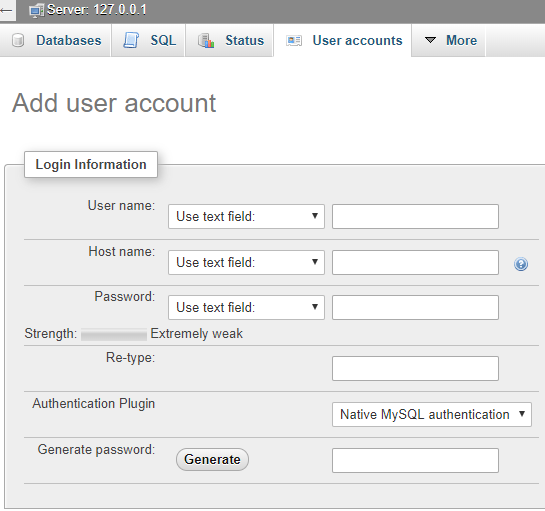
Step 4. In browser, go to localhost/phpmyadmin

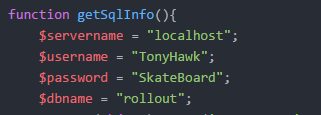


Step 5. Run both the rollout.sql and procedures.sql scripts provided in phpmyadmin



Step 6. Set the privilege of the database to one specified in image.





Step 7. Go to <http://localhost/DataBase/index.php> in browser.

Step 8. You can use any user from customer or administrator to log in. I recommend using username Admin and password Admin to test the queries in the admin side of site.

If you have any issues, feel free to contact me.

## Accomplishments:

For this project, I have accomplished a variety of tasks. First, I have designed a complicated ERD and schema. Then I implemented it into a web based application, by merging a functional a database and some web programming together. I can say that I have successfully integrated a database adequately into a web format. The project has help me gain useful skills in web programming and implementing a database online.

## Regular Queries:

Projection Query:

SELECT Customer\_ID, FirstName, $table FROM Customer

Selection Query:

SELECT Car\_ID, Make, Model, Mileage FROM Cars WHERE Mileage >= $min AND Mileage <= $max

Join Query:

SELECT c.Customer\_ID, c.FirstName, c.LastName, d.Car\_ID, d.make

FROM customer c, lease l, cars d

WHERE c.Customer\_ID = l.Customer\_ID

AND l.Car\_ID = d.Car\_ID;

Division Query:

SELECT c.Customer\_ID, c.FirstName

FROM Customer c

WHERE NOT EXISTS

(SELECT \* FROM Cars v

WHERE NOT EXISTS

(SELECT \* FROM scheduler s

WHERE s.Car\_ID = v.Car\_ID

AND s.Customer\_ID = c.Customer\_ID));

Aggregation Queries:

SELECT Car\_ID, Make, Model, Max(Mileage) FROM Cars

SELECT Count(\*) FROM Cars

Nested Aggregation Query:

SELECT VehicleType, AVG(mileage) FROM Cars GROUP BY VehicleType

Delete Operation Query:

DELETE FROM Customer WHERE Customer\_ID = '$user'

Update Operation:

UPDATE Cars SET itCondition = '$cond' WHERE Car\_ID = '$carid'

## Procedure Queries:

DELIMITER //

DROP PROCEDURE IF EXISTS myProc //

CREATE PROCEDURE myProc()

BEGIN

SELECT DISTINCT Make, Model, VehicleImage FROM cars;

END //

DELIMITER //

DROP PROCEDURE IF EXISTS myProc1//

CREATE PROCEDURE myProc1(p1 VARCHAR(255))

BEGIN

SELECT Customer\_ID, FirstName, p1 FROM Customer;

END //

DELIMITER //

DROP PROCEDURE IF EXISTS myProc2 //

CREATE PROCEDURE myProc2 (p1 VARCHAR(10))

BEGIN

SELECT DISTINCT Make, Model, VehicleImage FROM cars

WHERE VehicleType = p1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS profile//

CREATE PROCEDURE profile (param1 VARCHAR(10))

BEGIN

SELECT \* FROM customer

WHERE Customer\_ID = param1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS admin//

CREATE PROCEDURE admin (param1 VARCHAR(10))

BEGIN

SELECT \* FROM Administrator

WHERE StaffID = param1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS usedCar//

CREATE PROCEDURE usedCar (param1 VARCHAR(10))

BEGIN

SELECT d.Car\_ID, d.make, d.model

FROM customer c, lease l, cars d

WHERE c.Customer\_ID = l.Customer\_ID

AND l.Car\_ID = d.Car\_ID

AND c.Customer\_ID = param1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS leases//

CREATE PROCEDURE leases (param1 VARCHAR(10))

BEGIN

SELECT \*

FROM lease

WHERE Customer\_ID = param1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS bills//

CREATE PROCEDURE bills(param1 VARCHAR(10))

BEGIN

SELECT \*

FROM invoice

INNER JOIN billed ON invoice.InvoiceNum = billed.InvoiceNum

WHERE Customer\_ID = param1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS findCars //

CREATE PROCEDURE findCars (p1 VARCHAR(6))

BEGIN

SELECT Car\_ID, Make, Model, VehicleImage

FROM cars

WHERE placedLocation = p1

AND inUse = 0;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS used //

CREATE PROCEDURE used (p1 VARCHAR(6))

BEGIN

UPDATE cars

SET inUse = '1'

WHERE Car\_ID = p1;

END//

DELIMITER //

DROP PROCEDURE IF EXISTS scheduleAdd //

CREATE PROCEDURE scheduleAdd (p1 date, p2 time, p3 time, p4 int(11), p5 VARCHAR(10), p6 VARCHAR(6), p7 VARCHAR(6), p8 VARCHAR(6))

BEGIN

INSERT INTO Scheduler

VALUES(p1, p2, p3, p4, p5,p6, p7, p8);

END//

DELIMITER //

DROP PROCEDURE IF EXISTS del //

CREATE PROCEDURE del (p1 VARCHAR(10))

BEGIN

DELETE FROM Customers

WHERE Customer\_ID = p1;

END//