KARSHARE

CMPE 332 – FINAL REPORT

Abstract

Web-based application and backend database for carsharing venture in Kingston called Karshare. Karshare provides people with the convenience of a vehicle when they need it without having to purchase one and promote a healthier environment.

Marwan Galal

10076446

Table of Contents

INTRODUCTION 3

SUMMARY 3
ASSUMPTIONS 3

DATABASE DESIGN 4

ORIGINAL ER SCHEMA 4
FINAL ER SCHEMA 5
FINAL RELATIONAL SCHEMA 5
DATABASE CREATION 6

STATE MACHINE DIAGRAM AND SQL INTERACTIONS 8

STATE MACHINE DIAGRAM - USERS 8
SQL INTERACTIONS - USERS 9
LOGIN STATE 9
REGISTRATION STATE 10
CONSUMER VIEW STATE 11
VIEW CAR STATE 12
REQUEST BOOKING STATE 13
RESERVATIONS STATE 13
PICK-UP STATE 13
DROP-OFF STATE 14
CANCEL RESERVATION STATE 14
RENTAL HISTORY STATE 14
UPDATE PROFILE STATE 15
TERMINATE MEMBERSHIP STATE 15
STATE MACHINE DIAGRAM – EMPLOYEES/ADMINISTRATORS 16
SQL INTERACTIONS – EMPLOYEES/ADMINISTRATORS 17
DELETE MEMBER STATE 17
ADD CAR STATE 18
SHOW HISTORY STATE 18
RESERVATION STATUS STATE 19
MONTHLY INVOICE STATE 19
HIGH/LOW STATE 20
TODAY STATE 20
DAMAGED STATE 20
OVER 5000 STATE 21
RESPOND STATE 21

DISCUSSION 22

ENCOUNTERED PROBLEMS 22
IMPORTANT DESIGN AND IMPLEMENTATION DECISIONS 23
TECHNOLOGIES USED 23
DREAMWEAVER 23
SUBLIME TEXT 23
PHPMYADMIN 23
LOOKING BACK 23

USER GUIDE **24**

MEMBER **24**
LOGIN/LOGOUT 24
REGISTRATION 25
CONSUMER VIEW 26
VIEW CAR 27
REQUEST BOOKING 28
RESERVATIONS 29
PICK-UP 30
DROP-OFF 31
CANCEL RESERVATION 32
RENTAL HISTORY 32
UPDATE PROFILE 33
ADMINISTRATOR/EMPLOYEE **34**
ADMINISTRATION –HOME 34
DELETE MEMBER 35
ADD CAR 36
MONTHLY INVOICE 37
RENTAL HISTORY 38
RESERVATION STATUS OF ALL CARS IN FLEET 39
OVER 5000KM SINCE LAST MAINTENANCE 40
DAMAGED CARS 40
HIGHEST/LOWEST RENTALS 41
ALL RESERVATIONS FOR GIVEN DAY 41
RESPOND TO COMMENTS 42

INTRODUCTION

Summary

The project brief was to develop a web-based application and backend database for a carsharing venture called Karshare – name was changed from KTCS. A relational database management system was created, as well as a state machine diagram for the application, implementation using phpMyAdmin and MySQL, and scripting in PHP, CSS, HTML and JavaScript.

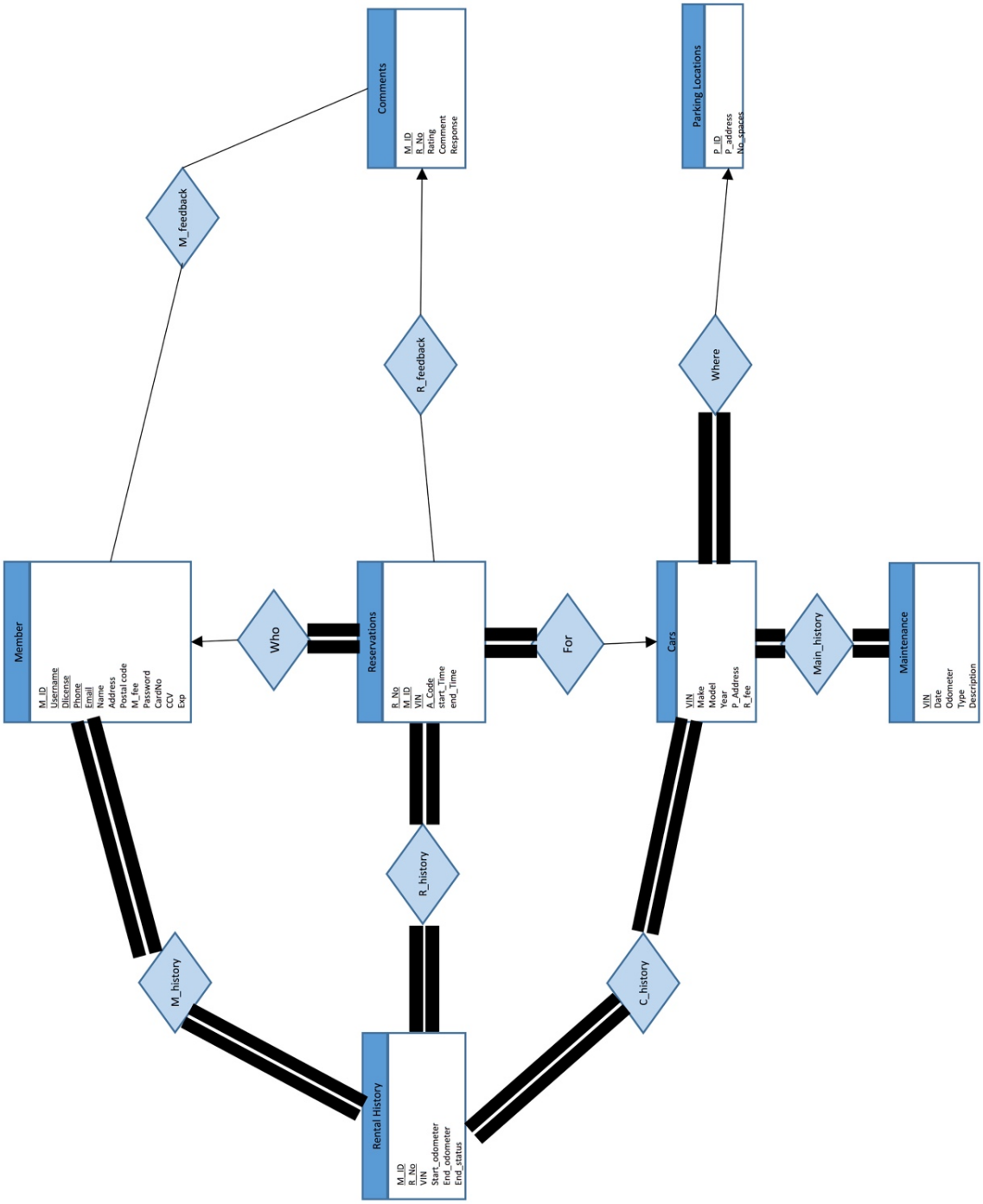
Assumptions

Along with criteria specified in the project brief, the following assumptions were made:

- Car can only be rented by one person at a time
- Cars can be reserved at any time of day
- Cars are picked up and dropped off at the same location
- Reservation number and access code are unique for each reservation to avoid problems with bookings
- Multiple cars can be seen for maintenance on the same day
- Members can only book cars for one day at a time

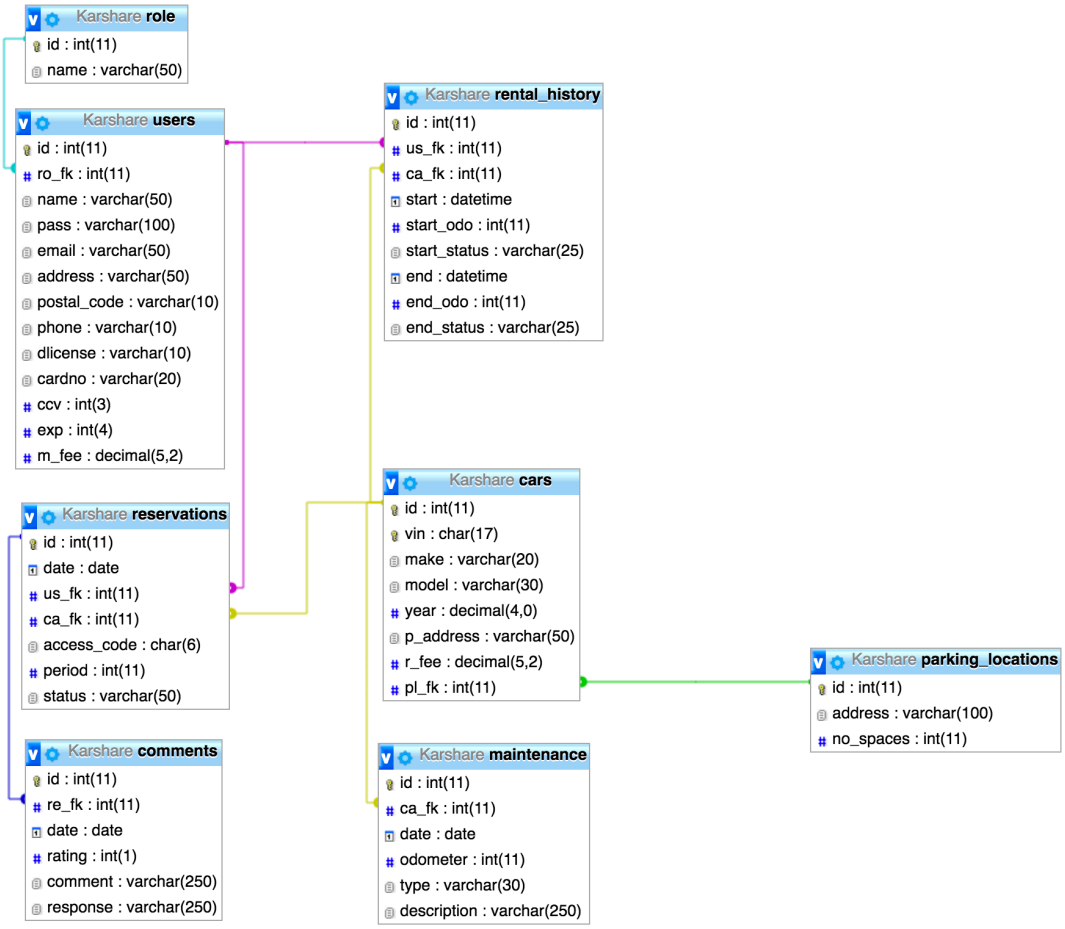
DATABASE DESIGN

Original ER Schema



Final ER Schema

The original database was redesigned to make it more relational and efficient. Foreign keys were relied on to implement this more relational design. Foreign key attribute are indicated in the schema by '*_fk'. Primary keys were reassigned, members table was renamed to users, attributes were added for functionality and a role relation was introduced to differentiate between users, employees and admin. The following is the enhanced ER schema generated via phpmyadmin, in UML notation.



Final Relational Schema

Role(id, name)
Users(id, ro_fk, name, pass, email, address, postal_code, phone, dlicense, cardNo, ccv, exp, m_fee)
Reservations(id, date, us_fk, ca_fk, access_code, period, status)
Comments(id, re_fk, date, rating, comment, response)
Rental_history(id, us_fk, ca_fk, start, start_odo, start_status, end, end_odo, end_status)
Cars (id, VIN, make, model, year, p_address, r_fee, pl_fk)
Parking_locations(id, address, no_spaces)
Maintenance (id, ca_fk, date, odometer, type, description)

Database Creation

To accompany the ER and relational schemas, and to provide further detail on the attributes, below is SQL implementation of database creation.

```sql
CREATE TABLE parking_locations
( id         INT PRIMARY KEY AUTO_INCREMENT,
  address    VARCHAR(100),
  no_spaces INTEGER);
```

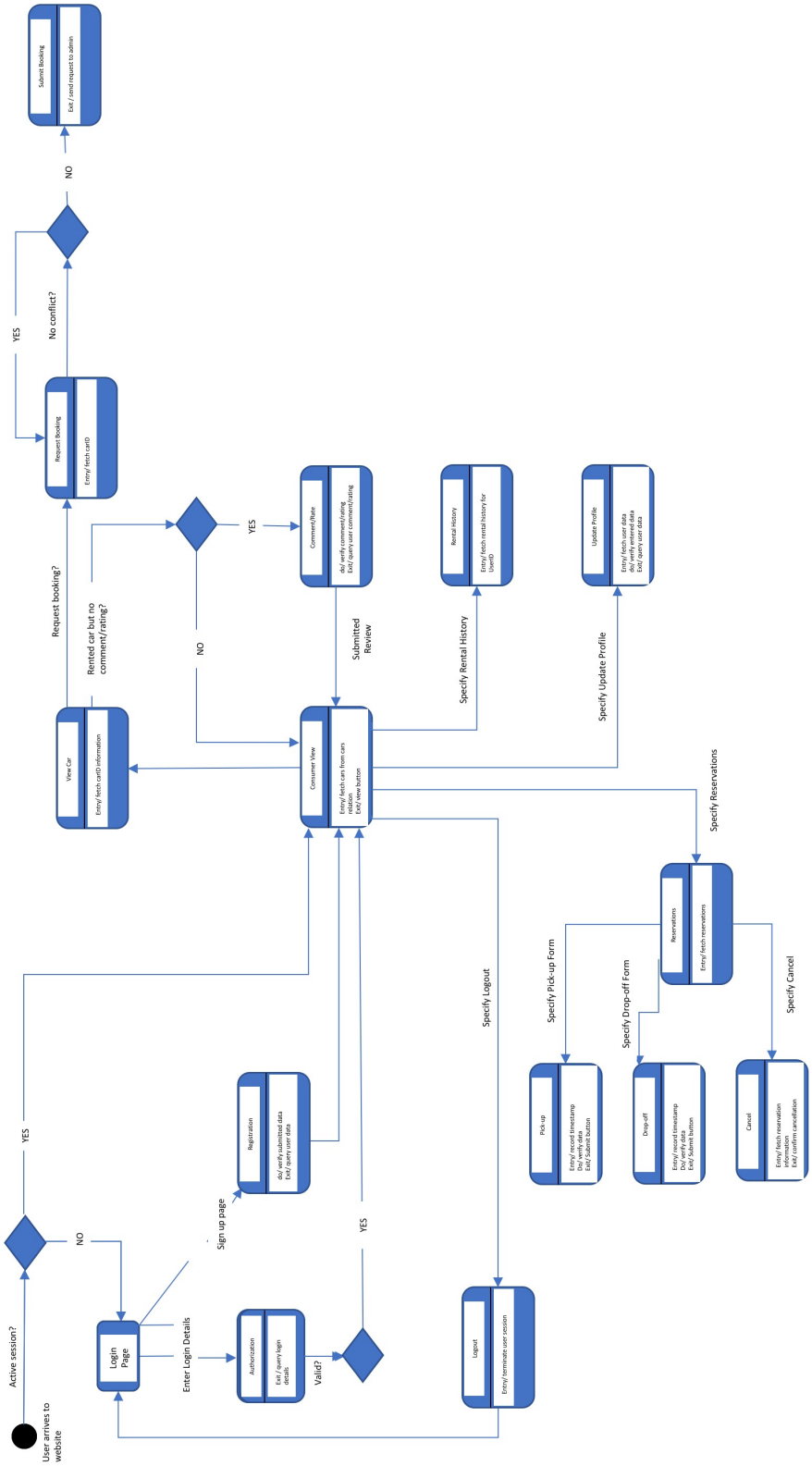
```sql
CREATE TABLE role (
  id INT          PRIMARY KEY AUTO_INCREMENT,
  name  VARCHAR(50)
);
```

```sql
CREATE TABLE users (
  id          INT          PRIMARY KEY AUTO_INCREMENT,
  ro_fk       INT          DEFAULT 1,
  name        VARCHAR(50),
  pass        VARCHAR(100),
  email       VARCHAR(50) NOT NULL,
  address     VARCHAR(50),
  postal_code VARCHAR(10),
  phone       VARCHAR(10),
  dlicense    VARCHAR(10) NOT NULL,
  cardno      VARCHAR(20),
  ccv         INT(3),
  exp         INT(4),
  m_fee       DECIMAL(5, 2),
  CONSTRAINT users_role_id_fk FOREIGN KEY (ro_fk) REFERENCES role (id)
);

CREATE TABLE cars
(
  id        INT PRIMARY KEY AUTO_INCREMENT,
  vin       CHAR(17)        DEFAULT '1GKDT13S162128177',
  make      VARCHAR(20),
  model     VARCHAR(30),
  year      NUMERIC(4, 0),
  p_address VARCHAR(50),
  r_fee     DECIMAL(5, 2),
  pl_fk     INT,
  CONSTRAINT cars_parking_locations_id_fk FOREIGN KEY (PL_FK) REFERENCES parking_locations (ID)
);
```

```sql
CREATE TABLE reservations
(
    id INT PRIMARY KEY AUTO_INCREMENT,
    date DATE,
    us_fk INT,
    ca_fk INT,
    access_code CHAR(6) NOT NULL,
    period INT,
    status VARCHAR(50),
    CONSTRAINT reservations_users_id_fk FOREIGN KEY (US_FK) REFERENCES users (id),
    CONSTRAINT reservations_cars_id_fk FOREIGN KEY (CA_FK) REFERENCES cars (id)
);

CREATE TABLE rental_history
(
  id           INT PRIMARY KEY AUTO_INCREMENT,
  us_fk        INT,
  ca_fk        INT,
  start   DATETIME DEFAULT '1900-01-01 00:00:00',
  start_odo    INT NOT NULL,
  start_status VARCHAR(25),
  end     DATETIME DEFAULT '4000-01-01 00:00:00',
  end_odo      INT NOT NULL,
  end_status   VARCHAR(25),
  CONSTRAINT rental_history_users_id_fk FOREIGN KEY (US_FK) REFERENCES users (id),
  CONSTRAINT rental_history_cars_id_fk FOREIGN KEY (CA_FK) REFERENCES cars (id)
);

CREATE TABLE maintenance
(
  id           INT PRIMARY KEY AUTO_INCREMENT,
  ca_fk        INT,
  date         DATE,
  odometer     INT NOT NULL,
  type         VARCHAR(30),
  description VARCHAR(250),
  CONSTRAINT maintenance_cars_id_fk FOREIGN KEY (CA_FK) REFERENCES cars (id)
);

CREATE TABLE comments
(
  id        INT PRIMARY KEY AUTO_INCREMENT,
  re_fk      INT,
  date      DATE,
  rating    INT(1),
  comment   VARCHAR(250),
  response  VARCHAR(250),
  CONSTRAINT comments_reservations_id_fk FOREIGN KEY (RE_FK) REFERENCES reservations (id)
);
```

STATE MACHINE DIAGRAM AND SQL INTERACTIONS

State Machine Diagram - Users



SQL Interactions - Users

Login State

The login state allows users, employees and administrators to login to the website using their email and password. There is a verification parameter that relies on relation “roles” to restrict user login to the consumerView.php page and employee/admin to admin.php page.

SQL Query

```
$query1 = "SELECT u.id, u.email, u.pass password, r.name role
           FROM users u INNER JOIN role r ON r.id = u.ro_fk
           WHERE email = ? AND u.pass = ? ;";
```

Sample

```
SELECT u.id, u.email, u.pass password, r.name role
                  FROM users u INNER JOIN role r ON r.id = u.ro_fk
                  WHERE email = 'test@test.com' AND u.pass = 'testpass';
```

id	email	password	role
1	test@test.com	testpass	member

Registration State

The registration state allows new users to create an account to use the service.

SQL Query

```
$sql = "INSERT INTO users (email, name, pass, address, postal_code, phone, dlicense,cardNo, ccv, exp)
VALUES ( '$email_var',
         '$name_var',
         '$password_var',
         '$address_var',
         '$postalcode_var',
         '$phone_var',
         '$dlicense_var',
         '$cardNo_var',
         '$ccv_var',
         '$exp_var')";
```

Sample

```
INSERT INTO users (email, name, pass, address, postal_code, phone, dlicense,cardNo, ccv, exp)
            VALUES ( 'cmpe332@cmpe.com', 'Patrick Martin',  'cmpe332', 'Goodwin Hall',
                     'G8N7GY','6137478866','29379239273','1111333355557777','999','1217');
```

←T→	id	ro_fk	name	pass	email	address	postal_code	phone	dlicense	cardno	ccv	exp	m_fee
Edit Copy Delete	1	1	Frank Brown	testpass	test@test.com	test address	28282828	9998887777	K2M3R5	9999888877776666	111	216	30.23
Edit Copy Delete	2	1	Alice Wong	testpass1	test1@test1.com	test address1	55566677	1112223333	K2M3R6	1111222233334444	222	115	30.23
Edit Copy Delete	3	1	Jack Harvey	testpass2	test2@test2.com	test address2	76767676	2223334444	K2M3R7	6666777755554444	333	217	30.23
Edit Copy Delete	4	2	James Arthus	testpassemp	testemp@test.com	test address	28282828	9998887777	K2M3R5	9999888877776666	111	216	30.23
Edit Copy Delete	7	1	Patrick Martin	cmpe332	cmpe332@cmpe.com	Goodwin Hall	G8N7GY	6137478866	2937923927	1111333355557777	999	1217	30.23

Consumer View State

When a user has logged in, they are redirected to the consumerView.php page. Here, the users can see all available cars in the fleet for rent today. The user can also search for specific criteria such as make, model, year, parking address, rental fee and a specific date.

SQL Query

```
$sql = "SELECT c.id, c.`make`, c.`model`, c.`year`, c.`P_address`, c.`r_fee`
        FROM `Cars` c
        LEFT JOIN `Reservations` r
        ON c.`id` = r.`ca_fk` AND DATE = r.`date`
        WHERE (
                (c.`make` LIKE '%$search%' ) or
                (c.`model` LIKE '%$search%') or
                (c.`year` LIKE '%$search%') or
                (c.`P_address` LIKE '%$search%') or
                (c.`r_fee` LIKE '%$search%') or
                (r. `date` LIKE '%$search%')
            )
            AND r.`id` IS NULL;";
```

```
$sql = "SELECT c.id, c.`make`, c.`model`, c.`year`, c.`P_address`, c.`r_fee`
        FROM `Cars` c
        LEFT JOIN `Reservations` r
        ON c.`id` = r.`ca_fk` AND DATE = r.`date`
            AND r.`id` IS NULL;";
```

Sample

id	make	model	year	P_address	r_fee
1	Ford	Focus	2010	test address	19.50
2	Chevrolet	Camaro	2012	test address	45.63
3	Cadillac	Escalade	2011	test address	101.50
4	dodge	challenger	1995	address1	56.00

View Car State

The view car state allows users to view more information on a specific car from the consumer view state. This includes make, model, year, car ID, as well as comments and ratings. Users can also comment and rate the car here. There are two SQL queries performed, along with PHP if-elseif, to differentiate between no comments and comments. Here, $addr_val is the car ID.

SQL Query

```
$sql1 = "SELECT VIN, make, model, year, p_address, r_fee, co.ID, Rating, comment
          FROM comments co
            INNER JOIN reservations re
              ON re.id = co.re_fk
            INNER JOIN cars c
              ON c.id = '$addr_var'";
$sql2 = "SELECT `Cars`.VIN, make, model, year, p_address, r_fee FROM `cars` WHERE `cars`.id = '$addr_var'";
```

Sample sql1

```
SELECT VIN, make, model, year, p_address, r_fee, co.ID, Rating, comment
                          FROM comments co
                            INNER JOIN reservations re
                              ON re.id = co.re_fk
                            INNER JOIN cars c
                              ON c.id = '1';
```

VIN	make	model	year	p_address	r_fee	ID	Rating	comment
1GKDT13S162128177	Ford	Focus	2010	test address	19.50	1	1	comment
1GKDT13S162128177	Ford	Focus	2010	test address	19.50	2	1	comment
1GKDT13S162128177	Ford	Focus	2010	test address	19.50	3	1	comment
1GKDT13S162128177	Ford	Focus	2010	test address	19.50	8	1	Bla bla

Sample sql2

```
SELECT `Cars`.VIN, make, model, year, p_address, r_fee FROM `cars` WHERE `cars`.id = '1'
```

VIN	make	model	year	p_address	r_fee
1GKDT13S162128177	Ford	Focus	2010	test address	19.50

Request Booking State

Users can request to book the car being viewed in the view car state for a given date and period.

SQL Query

```
$sql = "INSERT INTO `Reservations` (`us_fk`, `id`, `period`, `Status`, `ca_fk`, `Date`,) VALUES ('$val', NULL, '1', 'Pending', '$addy','$
  date_val');";
```

Sample

```
INSERT INTO `Reservations` (`us_fk`, `id`, `period`, `Status`, `ca_fk`, `Date`) VALUES ('1',
NULL, '1', 'Pending', '3','2017-04-03');
```

	id	date	us_fk	ca_fk	access_code	period	status
Edit Copy Delete	1	2017-01-01	1	3	123456	1	Cancelled
Edit Copy Delete	2	2017-02-01	1	2	654321	2	pending
Edit Copy Delete	3	2017-03-01	2	1	222333	3	pending
Edit Copy Delete	4	2017-04-03	3	4	1	1	Pending
Edit Copy Delete	5	2017-04-03	1	3	2	1	Pending

Reservations State

The reservation state allows users to see the reservations made and their status: accepted, rejected, cancelled or pending.

SQL Query

```
$sql = "SELECT * FROM Users NATURAL JOIN Reservations WHERE users.id ='$val'";
```

Sample

```
SELECT * FROM Users NATURAL JOIN Reservations WHERE users.id ='1';
```

id	ro_fk	name	pass	email	address	postal_code	phone	dlicense	cardno	ccv	exp	m_fee	date	us_fk	ca_fk	access_code	period	status
1	1	Frank Brown	testpass	test@test.com	test address	28282828	9998887777	K2M3R5	9999888877776666	111	216	30.23	2017-01-01	1	3	123456	1	Cancelled

Pick-up State

The pick-up state allows users to enter the required pickup information, including odometer reading, time (automatically entered) and car status: damaged, normal or not running. The data is entered into the rental_history relation because the reservation is no longer reserved.

SQL Query

```
$sql = "UPDATE rental_history SET start=NOW(), start_odo=$start_odometer_var, start_status=$status_var
 WHERE id='$val';";
```

Sample

```
UPDATE rental_history SET start= NOW(), start_odo= '10000', start_status= 'normal'
            WHERE id='1'
```

	id	us_fk	ca_fk	start	start_odo	start_status	end	end_odo	end_status
Edit Copy Delete	1	3	1	2017-04-11 08:19:12	10000	normal	2016-11-23 17:00:00	10010	damaged

Drop-off State

The drop-off state allows users to enter the required drop-off information, including odometer reading, time (automatically entered) and car status: damaged, normal, not running. The data is entered into the rental_history relation because the reservation is no longer reserved.

SQL Query

```
$sql = "UPDATE rental_history SET end = NOW(), end_odo=$end_odometer_var, end_status=$status_var
 WHERE id='$val';";
```

Sample

```
UPDATE rental_history SET end = NOW(), end_odo= '10020', end_status= 'damaged'
       WHERE id='1';
```

	id	us_fk	ca_fk	start	start_odo	start_status	end	end_odo	end_status
Edit Copy Delete	1	3	1	2017-04-11 08:19:12	10000	normal	2017-04-11 08:26:02	10020	damaged

Cancel Reservation State

The cancel reservation state allows a user to cancel their existing reservation.

SQL Query

```
$sql = "UPDATE Reservations SET Status='Cancelled' WHERE id='$RNo'";
```

Sample

```
UPDATE Reservations SET Status='Cancelled' WHERE id='1';
```

	id	date	us_fk	ca_fk	access_code	period	status
Edit Copy Delete	1	2017-01-01	1	3	123456	1	Cancelled

Rental History State

The rental history state allows users to view their rental history.

SQL Query

```
$sql = "SELECT * FROM Users NATURAL JOIN rental_history WHERE users.id ='$val'";
```

Sample

```
SELECT * FROM Users NATURAL JOIN rental_history WHERE users.id ='1';
```

id	ro_fk	name	pass	email	address	postal_code	phone	dlicense	cardno	ccv	exp	m_fee	us_fk	ca_fk	start	start_odo	start_status
1	1	Frank Brown	testpass	test@test.com	test address	28282828	9998887777	K2M3R5	9999888877776666	111	216	30.23	3	1	2017-04-11 08:19:12	10000	normal

end	end_odo	end_status
2017-04-11 08:26:02	10020	damaged

Update Profile State

In the update profile state, the user can make changes to any section of their account information. In order to allow for each section to be updated separately, separate SQL queries were made in separate PHP scripts for each section.

SQL Query

Here is an example of the password update section. Similarly, the user can do the same for any other users field, except for membership fee (m_fee).

```
$sql = "UPDATE `Users` SET pass='$Password_given' WHERE id='$val'";
```

Sample

```
1 UPDATE `Users` SET pass='testpass' WHERE id='1';
```

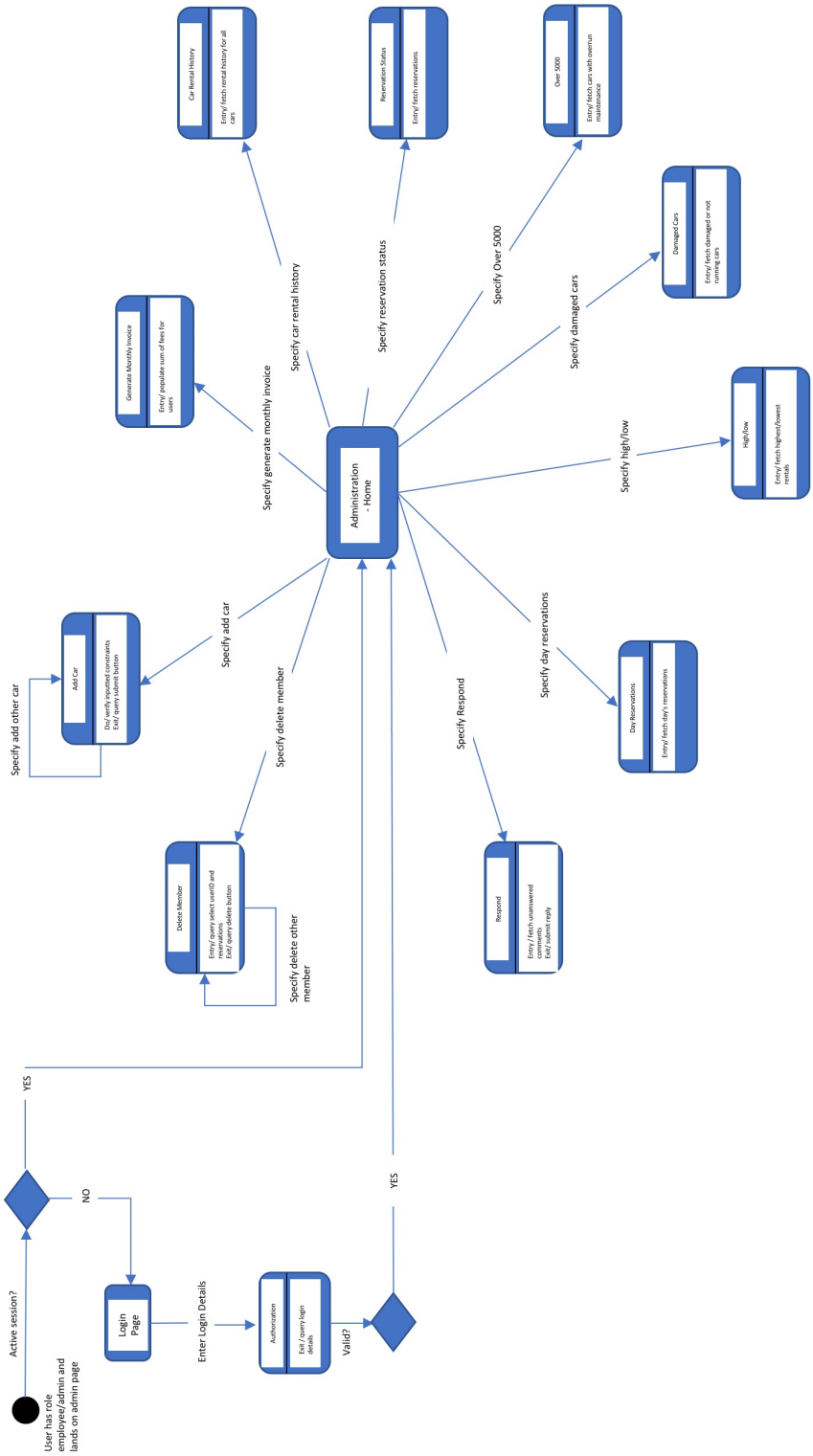
←T→	id	ro_fk	name	pass	email	addres
Edit Copy Delete	1	1	Frank Brown	testpass	test@test.com	test address

Terminate Membership State

This state allows users to delete their account and all associated reservations, given their user ID.

The queries are identical to the Delete Member State on the following page.

State Machine Diagram – Employees/Administrators



SQL Interactions – Employees/Administrators

Only users with the role “employee” or “admin” are able to access this portion of the website to add a layer of security.

Delete Member State

This state allows administrators to delete a member from the user list and all associated reservations, given a user ID.

SQL Query

```
$sql_1 = "DELETE FROM Users WHERE id = '$userID_given'";
$sql_2 = "DELETE FROM Reservations WHERE us_fk = '$userID_given'";
```

Sample

The user Patrick Martin, which was created to showcase the registration state functionality has been deleted from the users and reservations relations. The user’s rental history is saved for financial purposes.

```
1 DELETE FROM Users WHERE id = '7';
2 DELETE FROM Reservations WHERE us_fk = '7';
```

		id	ro_fk	name	pass	email
Edit Copy Delete		1	1	Frank Brown	testpass	test@test.com
Edit Copy Delete		2	1	Alice Wong	testpass1	test1@test1.com
Edit Copy Delete		3	1	Jack Harvey	testpass2	test2@test2.com
Edit Copy Delete		4	2	James Arthus	testpassemp	testemp@test.com

		id	date	us_fk	ca_fk	access
Edit Copy Delete		1	2017-01-01	1	3	123456
Edit Copy Delete		2	2017-02-01	1	2	654321
Edit Copy Delete		3	2017-03-01	2	1	222333
Edit Copy Delete		4	2017-04-03	3	4	1
Edit Copy Delete		5	2017-04-03	1	3	2

Add Car State

This state allows administrators to add a car to the fleet.

SQL Query

```
$sql = "INSERT INTO cars (VIN, make, model, year, p_address, r_fee)
VALUES ( '$VIN_var',
        '$make_var',
        '$model_var',
        '$year_var',
        '$paddress_var',
        '$rfee_var')";
```

Sample

```
INSERT INTO cars (VIN, make, model, year, p_address, r_fee)
      VALUES ( '$JDBSIJDO9237H',
            'Chevrolet',
            'Camaro',
            '2017',
            'address 3',
            '150');
```

		id	vin	make	model	year	p_address	r_fee
Edit Copy Delete		1	1GKDT13S162128177	Ford	Focus	2010	test address	19.50
Edit Copy Delete		2	WAUDH74F06N157747	Chevrolet	Camaro	2012	test address	45.63
Edit Copy Delete		3	1HTGGAHR71H398655	Cadillac	Escalade	2011	test address	101.5
Edit Copy Delete		4	kddksnd	dodge	challenger	1995	address1	56.00
Edit Copy Delete		5	$JDBSIJDO9237H	Chevrolet	Camaro	2017	address 3	150.0

Show History State

This state allows administrators to show all rental history for all cars.

SQL Query

```
$sql = "SELECT ca_fk, id, start_odo, end_odo, end_status FROM rental_history";
```

Sample

		ca_fk	id	start_odo	end_odo	end_status
Edit Copy Delete		1	1	10000	10020	damaged
Edit Copy Delete		2	2	330	450	normal
Edit Copy Delete		2	3	450	490	not running
Edit Copy Delete		2	4	10	330	normal
Edit Copy Delete		1	5	4000	10000	normal

Reservation Status State

This state allows administrators to view the parking location, reservation status and date of reservation of all reserved cars in fleet.

SQL Query

```
$sql = "SELECT pl.address, date, r.status, c.make, c.model, c.id, c.year
        FROM parking_locations pl INNER JOIN cars c ON pl.ID = c.pl_fk
        INNER JOIN reservations r ON c.id = r.CA_FK;";
```

Sample

address	date	status	make	model	id	year
testaddress2	2017-01-01	Cancelled	Cadillac	Escalade	3	2011
testaddress1	2017-02-01	pending	Chevrolet	Camaro	2	2012
testaddress1	2017-03-01	pending	Ford	Focus	1	2010
testaddress2	2017-04-03	Pending	Cadillac	Escalade	3	2011

Monthly Invoice State

This state allows administrators to generate a monthly invoice for all users.

SQL Query

```
$sql = "SELECT DATE('2016-12-01') report_month, name, email, address, postal_code,
        phone, m_fee + coalesce(rf.r_fee, 0) total_fee
        FROM users u LEFT JOIN (
        SELECT
        rh.us_fk, sum(c.r_fee) r_fee
        FROM rental_history rh INNER JOIN cars c ON rh.ca_fk = c.id
        WHERE DATE'2016-12-01' between date_sub(start, INTERVAL dayofmonth(start) -1 DAY)
        and date_sub(end, INTERVAL dayofmonth(start) -1 DAY)
        GROUP BY rh.us_fk) rf ON rf.us_fk = u.id;";
```

Sample

report_month	name	email	address	postal_code	phone	total_fee
2016-12-01	Frank Brown	test@test.com	test address	28282828	9998887777	30.23
2016-12-01	Alice Wong	test1@test1.com	test address1	55566677	1112223333	75.86
2016-12-01	Jack Harvey	test2@test2.com	test address2	76767676	2223334444	75.86
2016-12-01	James Arthus	testemp@test.com	test address	28282828	9998887777	30.23

High/Low State

This state allows administrators to view the best and worst performing cars in the fleet.

SQL Query

```
$sql1 = "SELECT c.model, coalesce(rh.total_rent, 0) total_rent, c.make, c.id, c.year
         FROM cars c LEFT JOIN ( SELECT ca_fk, count(*) total_rent FROM rental_history GROUP BY ca_fk) rh
         ON rh.ca_fk = c.id
         ORDER BY total_rent DESC
         LIMIT 1";
$sql2 = "SELECT c.model, coalesce(rh.total_rent, 0) total_rent, c.make, c.id, c.year
         FROM cars c LEFT JOIN ( SELECT ca_fk, count(*) total_rent FROM rental_history GROUP BY ca_fk) rh
         ON rh.ca_fk = c.id
         ORDER BY total_rent
         LIMIT 1";
```

Sample

model	total_rent	make	id	year
Camaro	3	Chevrolet	2	2012

model	total_rent	make	id	year
Camaro	0	Chevrolet	5	2017

Today State

Allows administrators to view all of a given day's reservations.

SQL Query

```
$sql = "SELECT r.ID, r.date, r.period, r.status, c.id, c.vin, c.make, c.model, c.year
        FROM reservations r
            INNER JOIN cars c
              ON c.id = r.CA_FK
        WHERE 1 = 1
          AND r.date = DATE'2017-01-01'
        ;";
```

Sample

ID	date	period	status	id	vin	make	model	year
1	2017-01-01	1	Cancelled	3	1HTGGAHR71H398655	Cadillac	Escalade	2011

Damaged State

This state allows administrators to view all damaged or not running cars.

SQL Query

```
$sql = "SELECT r.ca_fk, r.end_status, c.make, c.model, c.year
        FROM rental_history r, cars c
        WHERE c.id = r.ca_fk AND (end_status = 'damaged' OR end_status = 'not running');";
```

Sample

ca_fk	end_status	make	model	year
1	damaged	Ford	Focus	2010
2	not running	Chevrolet	Camaro	2012

Over 5000 State

Allows administrators to view all cars that have run over 5000km since last maintenance.

SQL Query

```
$sql = "SELECT c.id, c.vin, c.make, c.model, c.year, rh.end_odo - coalesce(m.last_odo, 0) odo_since_maintenance
        FROM cars c
          LEFT JOIN (SELECT ca_fk, (odometer) last_odo FROM maintenance group by ca_fk) m
            ON m.CA_FK = c.id
          INNER JOIN (SELECT ca_fk, (end_odo) end_odo FROM rental_history group by ca_fk) rh
            ON rh.ca_fk = c.id
        WHERE (rh.end_odo - m.last_odo) > 5000
        ;";
```

Sample

id	vin	make	model	year	odo_since_maintenance
1	1GKDT13S162128177	Ford	Focus	2010	7020

Respond State

This state allows administrators to check if comments have not been responded to and allows for response to comments in PHP.

SQL Query

```
$sql = "SELECT co.date, rating, comment, response, u.email, u.address, u.name, re.ID, re.Date, re.status
        FROM comments co
          INNER JOIN reservations re
            ON re.id = co.re_fk
          INNER JOIN users u
            ON u.id = re.US_FK
        WHERE co.response IS NULL;";
```

Sample

date	rating	comment	response	email	address	name	ID	Date	status
2017-04-10	1	Bla bla	*NULL*	test@test.com	test address	Frank Brown	1	2017-01-01	Cancelled

Discussion

Encountered Problems

The major problem encountered was the lack of experience I had in php, HTML/CSS and JavaScript. Therefore, I used a lot of online resources, the course lecture on php and trial and error to get the website functioning. As a result of the little-to-no experience I had with the scripting languages, I used W3Schools for material, online forums such as stackoverflow and an adhoc method of putting together open source scripts, and editing them accordingly. This was extremely time consuming as I would have to read, understand and tailor the code to this application. However, it was efficient in that I did not need to learn three or four different scripting languages before beginning the web-application.

I was unable to find a group to partner with and thus I was forced to work alone. While I was able to learn a lot more, and implement the course material myself, it was a drawback not to have members to rely on, brainstorm with or with differing levels of experience. As a result, many tasks were completed past the expected timeframe.

I chose to reassemble the ER schema in a more relational – less redundant – manner, and thus some of the SQL queries became quite difficult to implement, with multiple joins. Furthermore, a few new functions were discovered via stackoverflow and W3Schools to implement more complex states such as monthly invoice, high/low and over 5000 states. Furthermore, due to the relational aspect of the database, I had to keep track of many different variables and how they are related. However, after a couple of queries, the database sunk in and it became natural to code the queries. Finally, when creating the UML statechart diagram, I was unable to find any free resources to create one. As a result, Microsoft Powerpoint was used, although it is not used make particularly sophisticated diagrams.

Major problems occurred when coding in php, usually due to incorrect syntax and customs. I wanted the website to be dynamic and less brute force through the scripts and so I relied on php to provide that dynamic, user-operated environment. In most cases I was able to do so, however in a couple of admin scripts, such as monthly invoice and the today states, I relied on a manual entry method into the script. The biggest dynamic problem with the php scripts was passing variables between independent scripts. A hyperlink reference was used to include the passed variable, and this solved the problem by using $_POST(Address) to retrieve the passed variable. Furthermore, there was a major issue in connecting to the database on each page. This resulted in an error that would not allow me to even view the page. I took the decision to power forward and finish all the functionalities in all the scripts before fixing the connection issues, so I was blindly coding through. The reason I chose to do so was because I knew the connection error was the same amongst each script and I was careful with scripting the remaining functionality to avoid errors. In the end, there were a few minor errors that were fixed after the connection issue. Finally, it was difficult to add the security to separate admin and user login to prevent users from accessing the administration home. However, after some research, the correct php syntax was implemented and the added security achieved.

There were a couple of issues running the server as XAMPP would not launch them; it would attempt to start but ultimately fail. After some attempts to fix the issue through terminal, I chose to uninstall and reinstall it, which solved the issue.

Ultimately, by working through and debugging each script, I was able to achieve a fully functioning website.

Important Design and Implementation Decisions

After completion of the first phase of the project, database design, I reviewed the feedback. I was told that I had some unnecessary constraints, and too many primary keys. As a result, I chose to spend a few hours revamping the design to make it more relational and less redundant.

Using a list of the required functionality of the website, a logical flow was determined and the state diagram was completed. I had to go through a couple of iterations of it as I was implementing scripts, however it remained similar to the original design.

It was important that the website be fully functional and as dynamic as possible for a swift user experience. As a result, most of the effort was placed into SQL and PHP, with HTML/CSS and JavaScript deemed less important. Therefore, a simple CSS design was selected.

Technologies Used

Dreamweaver

Since I was unable to view the website due to the connection error in the PHP script, I relied on Adobe Dreamweaver to give a real-time depiction of the design, and to provide easy fixing of HTML/CSS.

Sublime Text

Sublime text is my preferred text editor for code because it provides a colour-coded representation of variables and different scripting languages. This meant that minor errors, like a missing character, were quickly identified and fixed.

phpMyAdmin

phpMyAdmin was used to access the database management system. It was used to test and debug SQL queries, it provided a visual view of relations and various joins, and it provided a sophisticated ER diagram.

Looking Back

Looking back I would spend more time learning HTML and PHP, as that would have sped up the development process and led to a more sophisticated final product. Furthermore, I always prefer to work in a team, so that would have been beneficial in speed of development and perhaps enhanced functionality. Finally, I would have used a debugger for php instead of trying to fix errors that would appear on the website.

User Guide

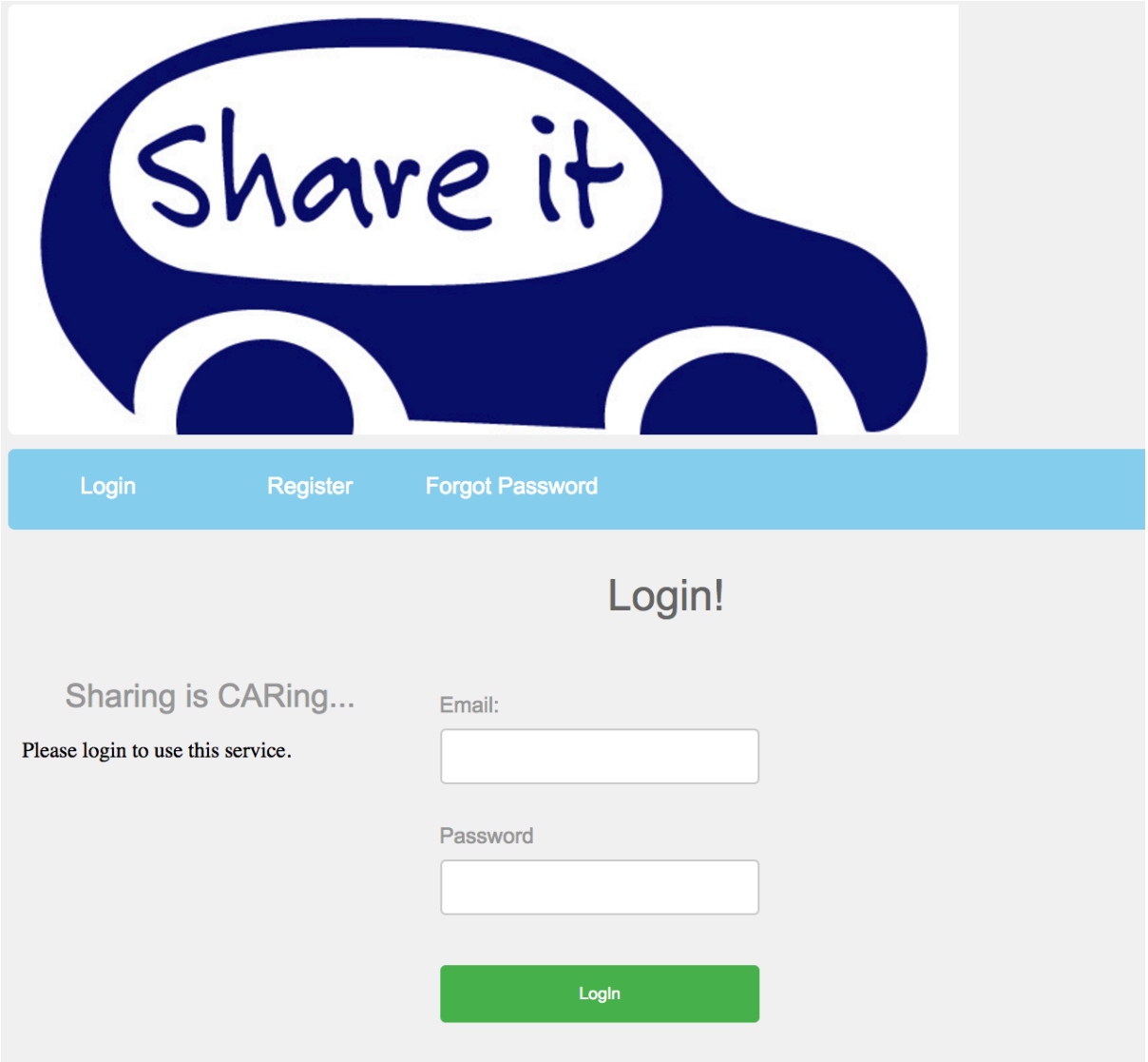
This guide serves as a manual to properly traversing the website and making use of all of its functionalities. It has been split into a user section and an administrator/employee section.

Member

To begin, open your browser and type in localhost/Karshare/Login.php

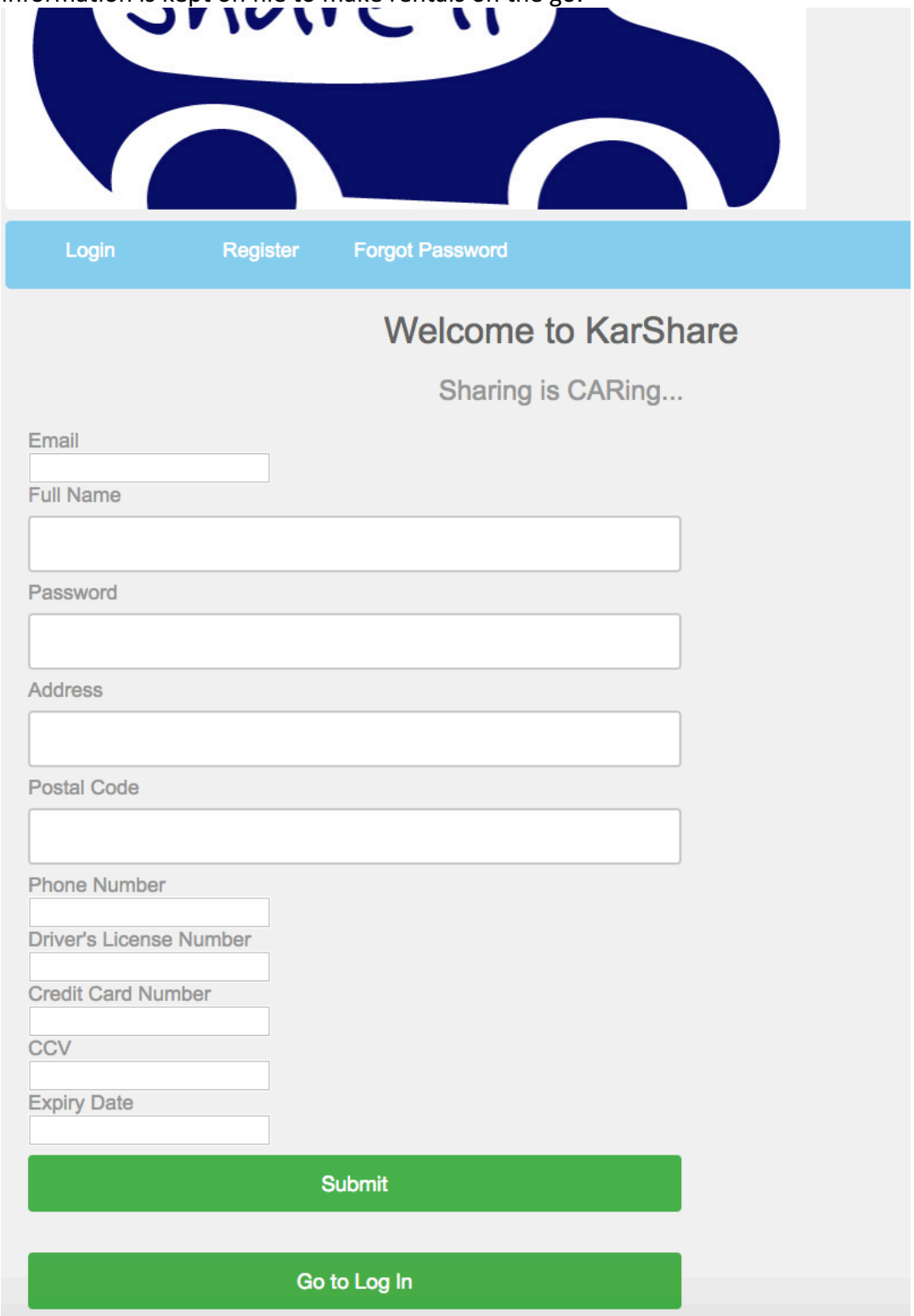
Login/logout

When the web address has been entered, you will arrive at the login page. Existing members can login using their email and password. New members can click on the register button to create an account. At any time after login, you can logout by clicking the logout button in the navigation bar – this will redirect you to the login page.



Registration

If you are a new member, please fill out your details and click on the Submit button. Once submitted, you can click on Go To Login to redirect you to the login page. Your credit card information is kept on file to make rentals on the go.



Consumer View

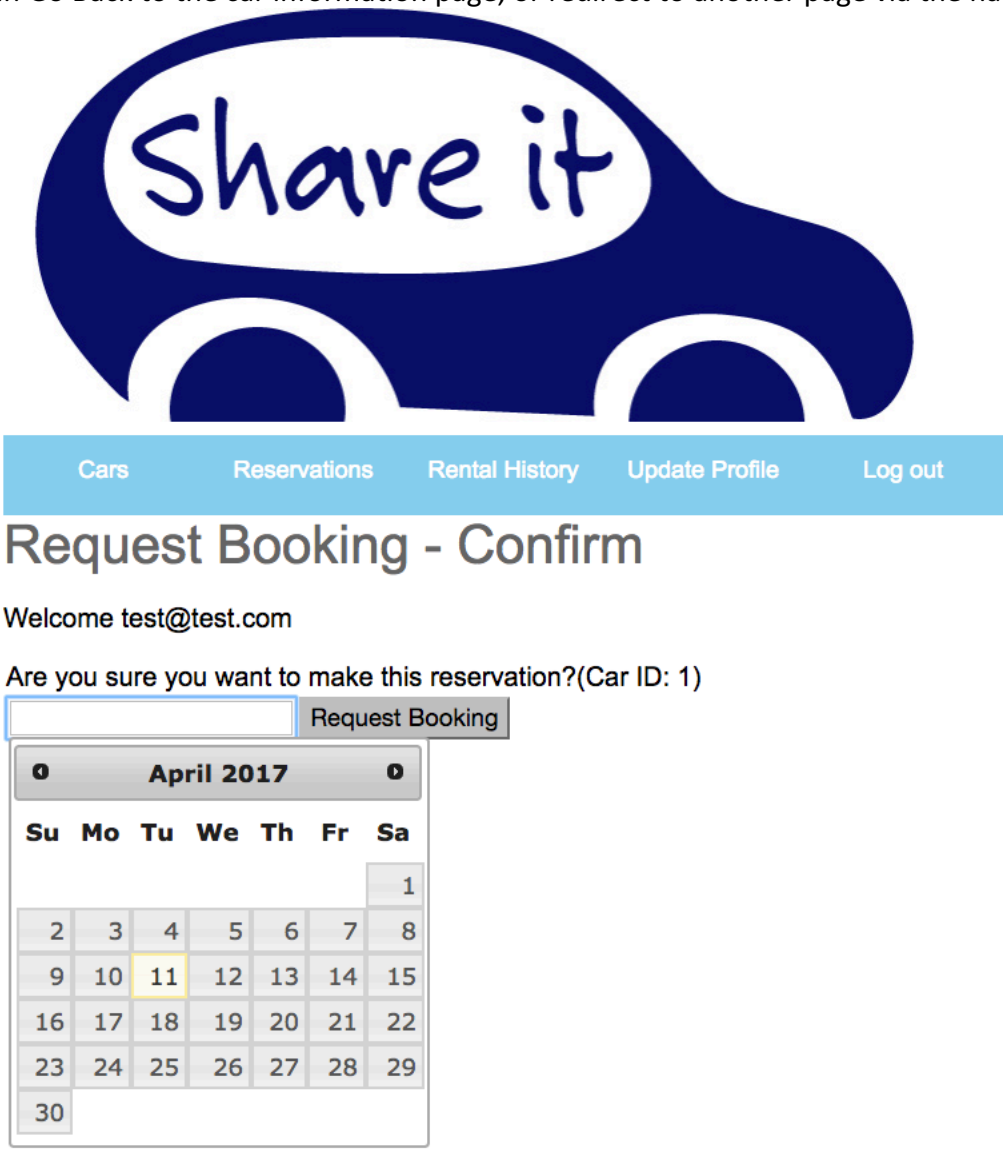
If your login is successful, you will arrive at the consumer view page. Here you can view all of the cars in the fleet. You can also search cars with specific information, such as make, model, year, date of reservation, rental fee and parking location. You can redirect to any of the pages in the navigation bar, or you can click on View to view more information about a specific car. Finally, if you would like to terminate your membership, you can click on the 'terminate membership' button.

View Car

If you click View next to a specific car, you will be taken to the page of specific information on it. Here, you will be able to view basic information, comments and ratings, add a comment or rating if you have used this car before and finally move forward with requesting a booking.

Request Booking

On the following page you will be able to select a date for your reservation. Otherwise, you can Go Back to the car information page, or redirect to another page via the navigation bar.

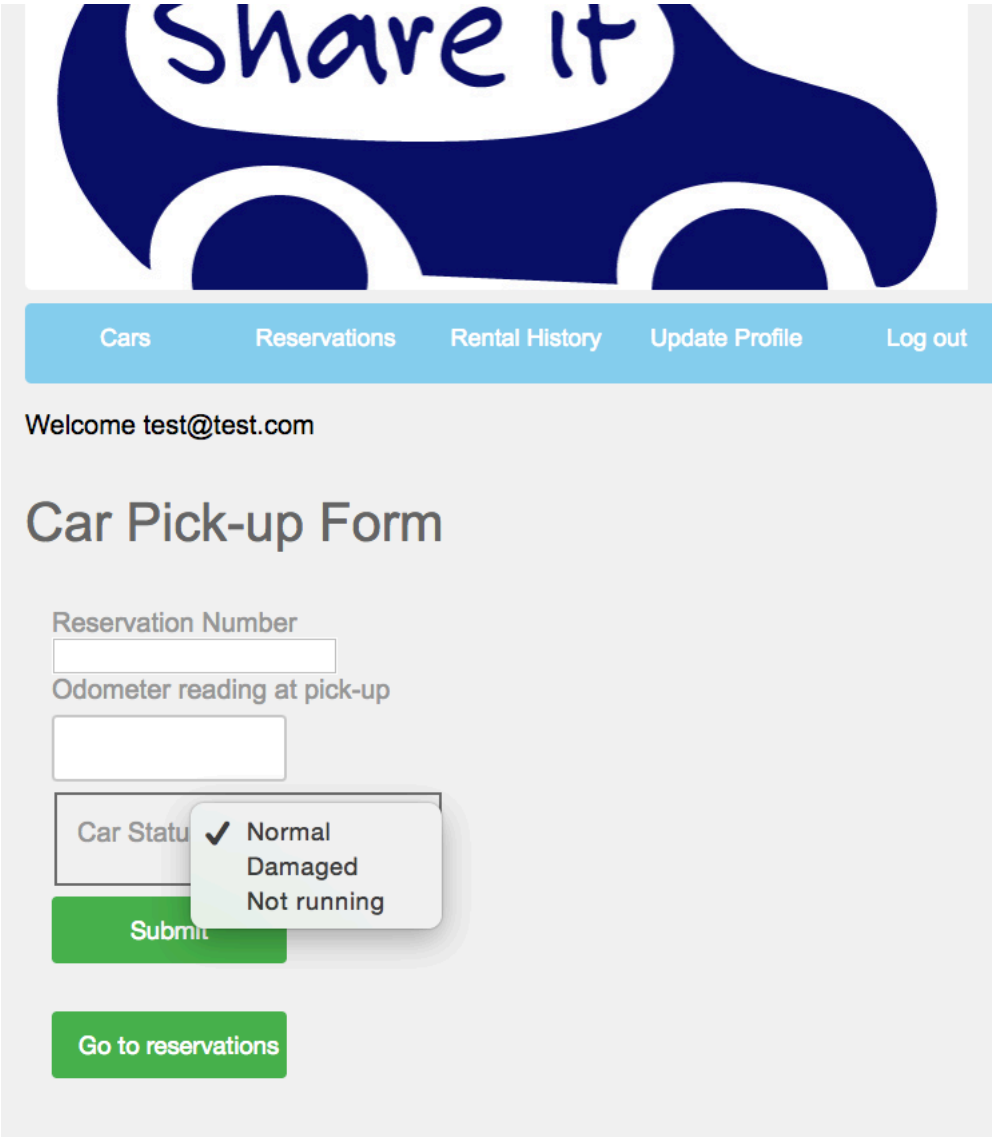


Reservations

Clicking on the Reservations link will take you to a page showing your pending and cancelled reservations. From here, you can click on Pick-up From to fill in the required pick-up information, Drop-off form for the required drop-off information and Cancel to cancel a pending reservation.

Pick-up

When picking up a car, each user is required to fill in this pick-up form that asks for the reservation number, odometer reading and status at pick-up. As soon as you click on Submit, your reservation will begin. Please be honest! Dishonesty will lead to member deletion.

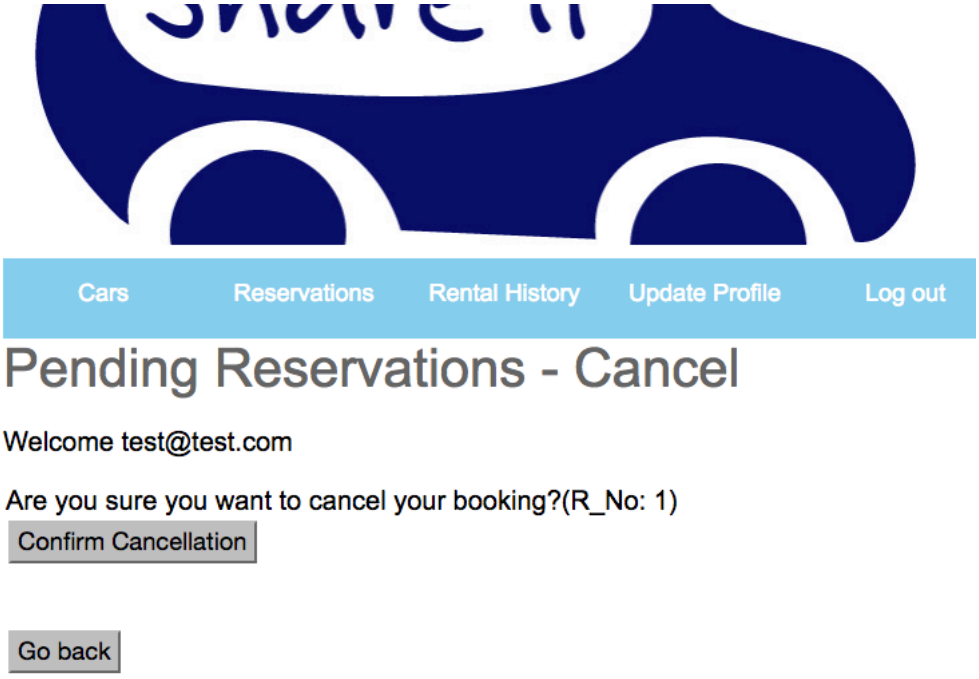


Drop-off

Similar to the pick-up form, the drop-off form requests the same information. As soon as you click on Submit, your reservation will end. From here, you can Go to Rental History to see the saved information. Please be honest! Dishonesty will lead to member deletion.

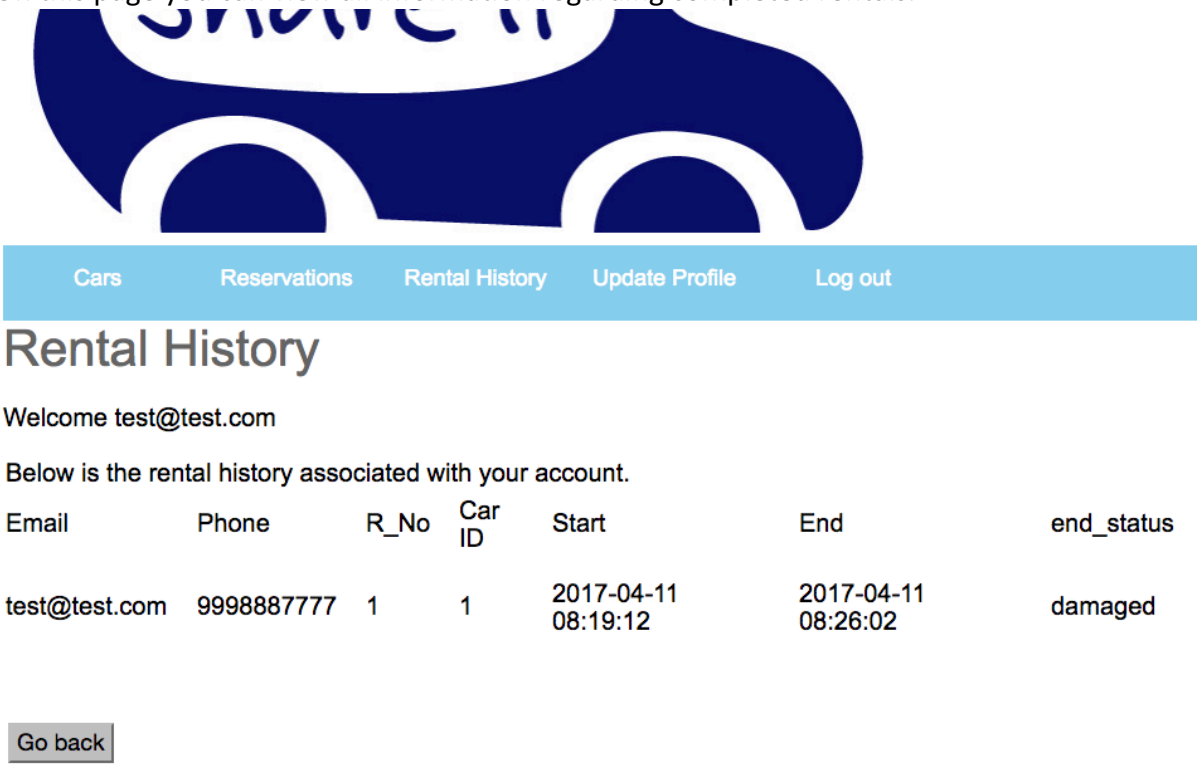
Cancel Reservation

If you would like to cancel your reservation, you will be asked to confirm, or you can Go back.



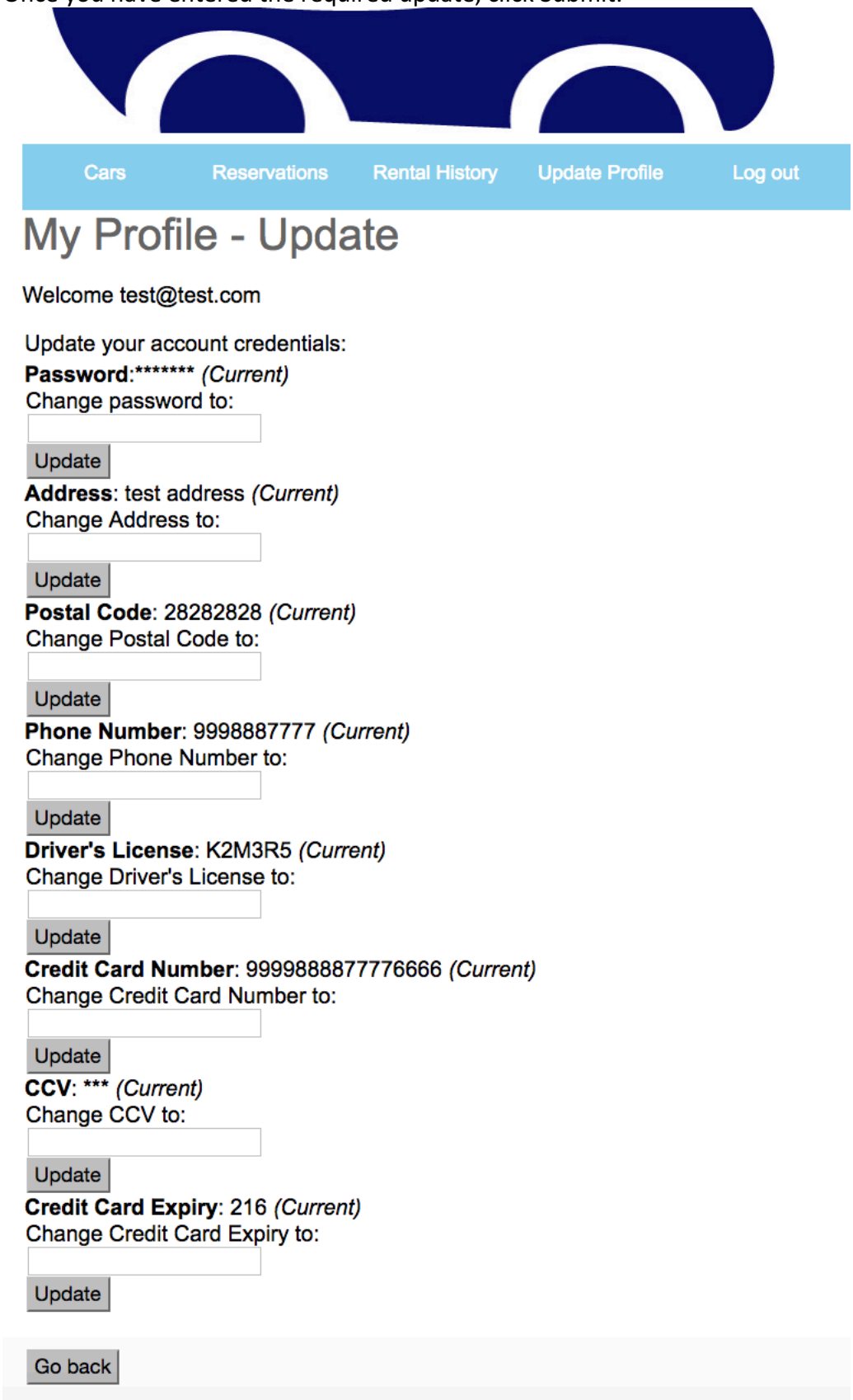
Rental History

On this page you can view all information regarding completed rentals.



Update Profile

On the update profile page you can change one or more sections of your information on file. Your current information is also shown, except for passwords and CCV for security reasons. Once you have entered the required update, click Submit.

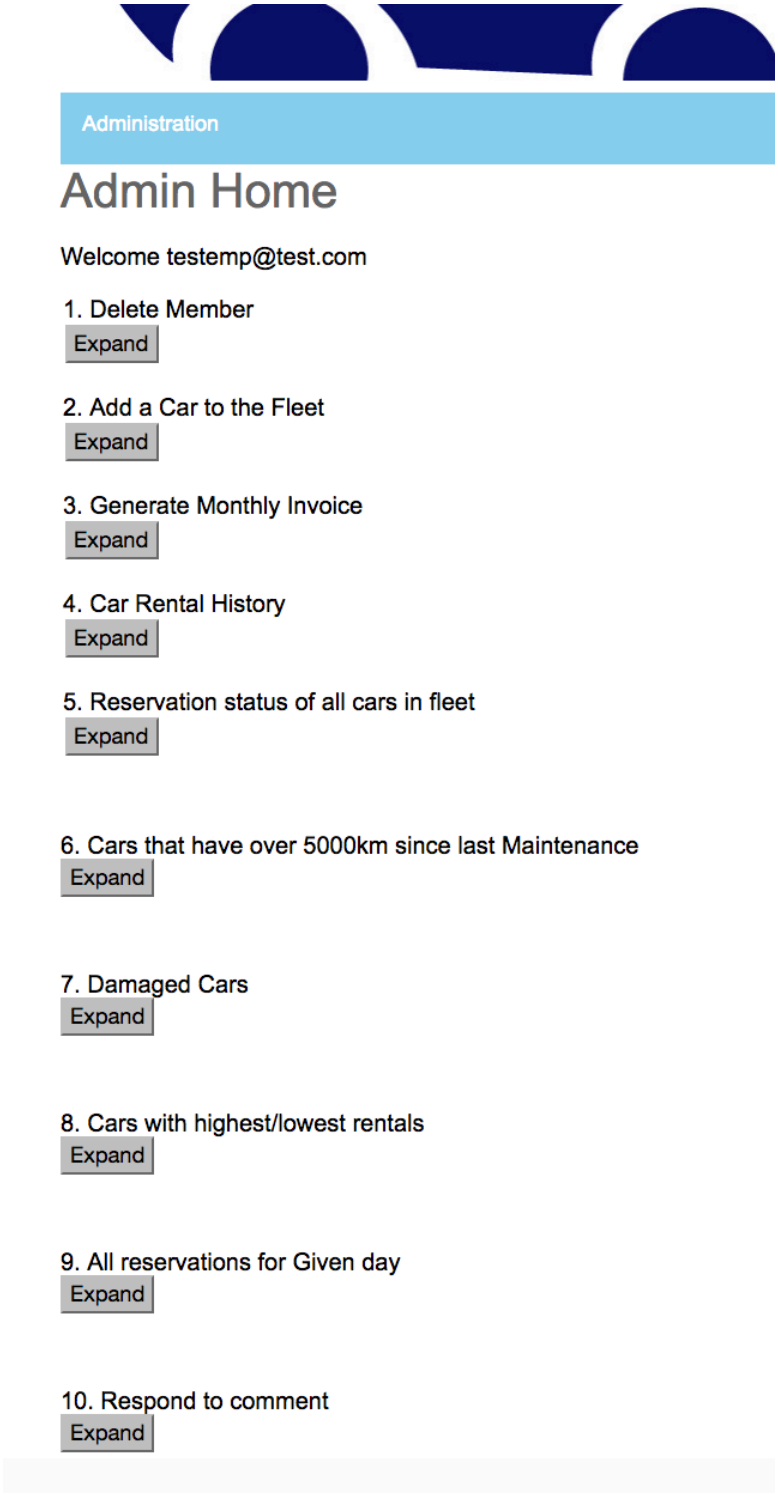


Administrator/Employee

Open your browser to localhost/Karshare/Login.php. Type in your login details and you will be redirected to the Administration – Home.

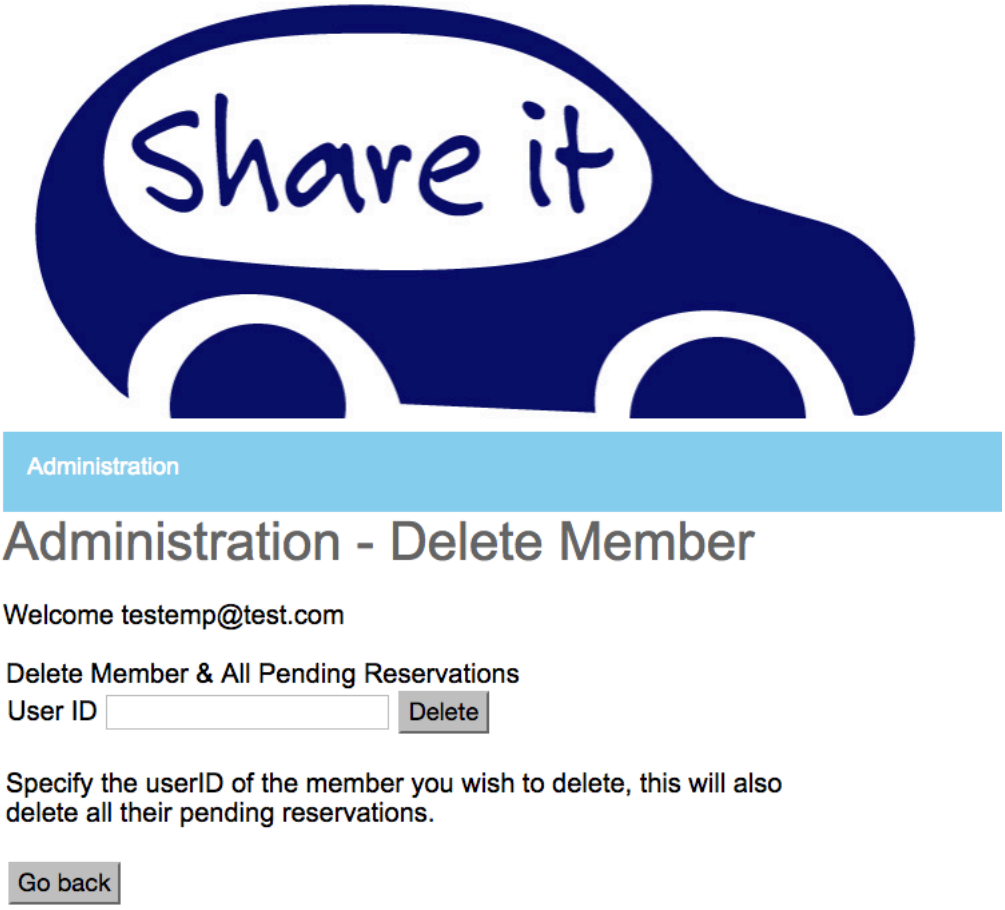
Administration –Home

Here is the homepage for administrators and employees. You will be able to navigate through the listed administrative functions.



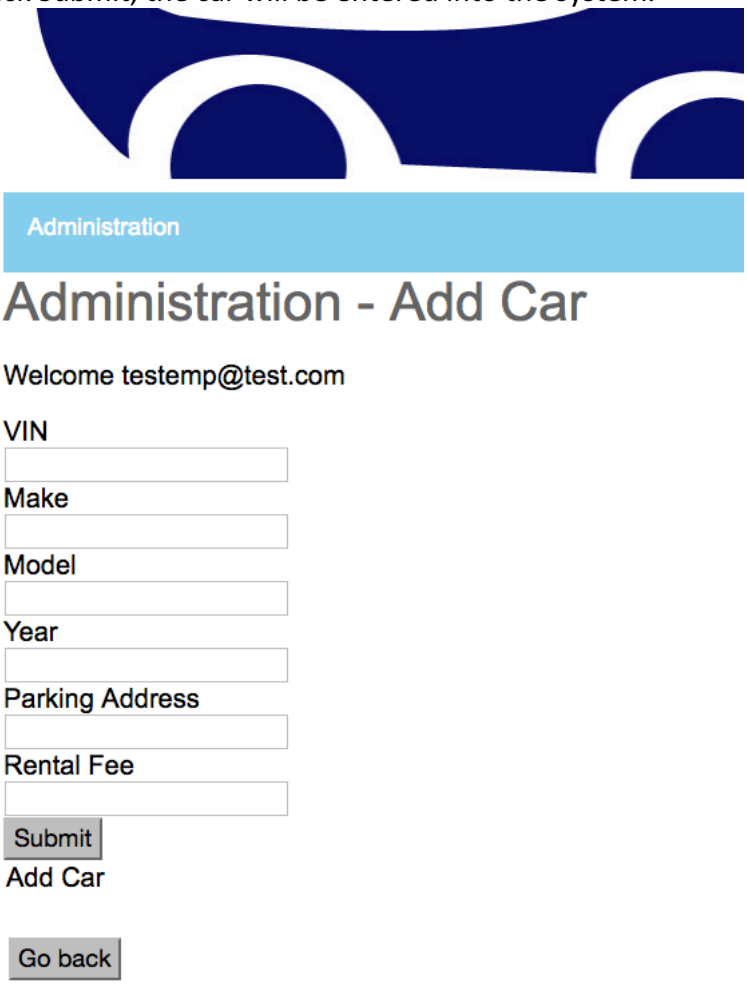
Delete Member

On this page you can enter a User ID. Upon clicking Delete, that user and all of their pending reservations will be deleted. You can go back to the Administration – Home by clicking on Administration or Go back.



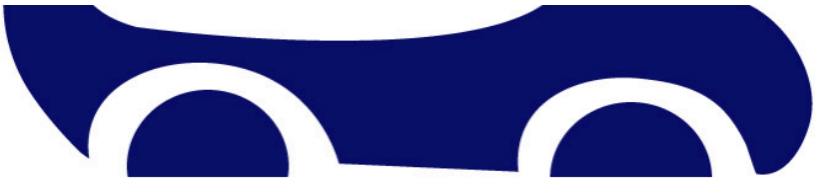
Add Car

On this page, you will be able to add a car to the fleet. Enter the required information and click Submit, the car will be entered into the system.



Monthly Invoice

On this page, you will be able to generate a monthly invoice for all users. Each month a system administrator will change the script so that you are only viewing invoices for the current month.



Administration

Administration - Monthly Invoice

Welcome testemp@test.com

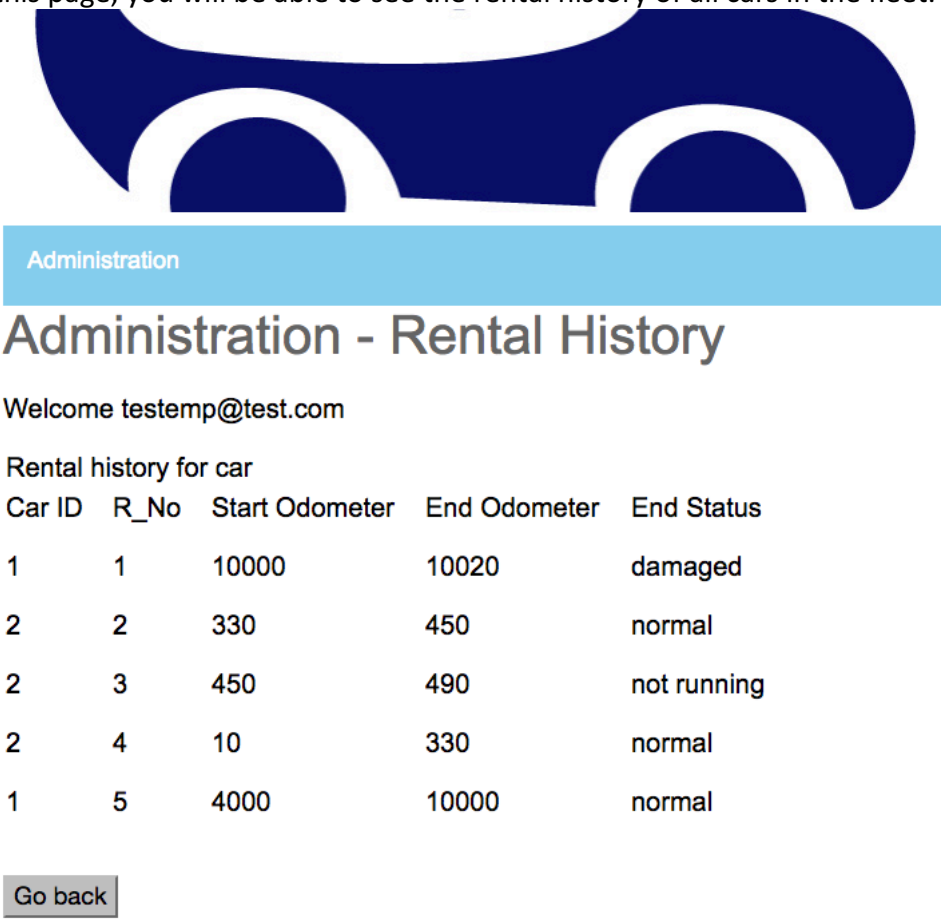
Generate a monthly invoice for a member

Month	Name	Email	Address	Postal Code	Phone	Total
2016-12-01	Frank Brown	test@test.com	test address	28282828	9998887777	30.23
2016-12-01	Alice Wong	test1@test1.com	test address1	55566677	1112223333	75.86
2016-12-01	Jack Harvey	test2@test2.com	test address2	76767676	2223334444	75.86
2016-12-01	James Arthus	testemp@test.com	test address	28282828	9998887777	30.23

Go back

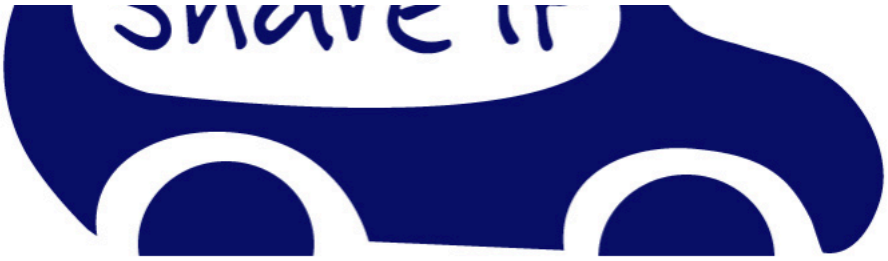
Rental History

On this page, you will be able to see the rental history of all cars in the fleet.



Reservation Status of all cars in fleet

For each parking location, you will be able to see the reservations and dates for all cars in the fleet. All other cars are available.



Administration

Administration - Reservation Status by Parkir Location

Welcome testemp@test.com

View reservation status and date of reservation of all reserved cars in fleet

Address	Status	Date	Car ID	Make	Model	Year
testaddress2	Cancelled	2017-01-01	3	Cadillac	Escalade	2011
testaddress1	pending	2017-02-01	2	Chevrolet	Camaro	2012
testaddress1	pending	2017-03-01	1	Ford	Focus	2010
testaddress2	Pending	2017-04-03	3	Cadillac	Escalade	2011

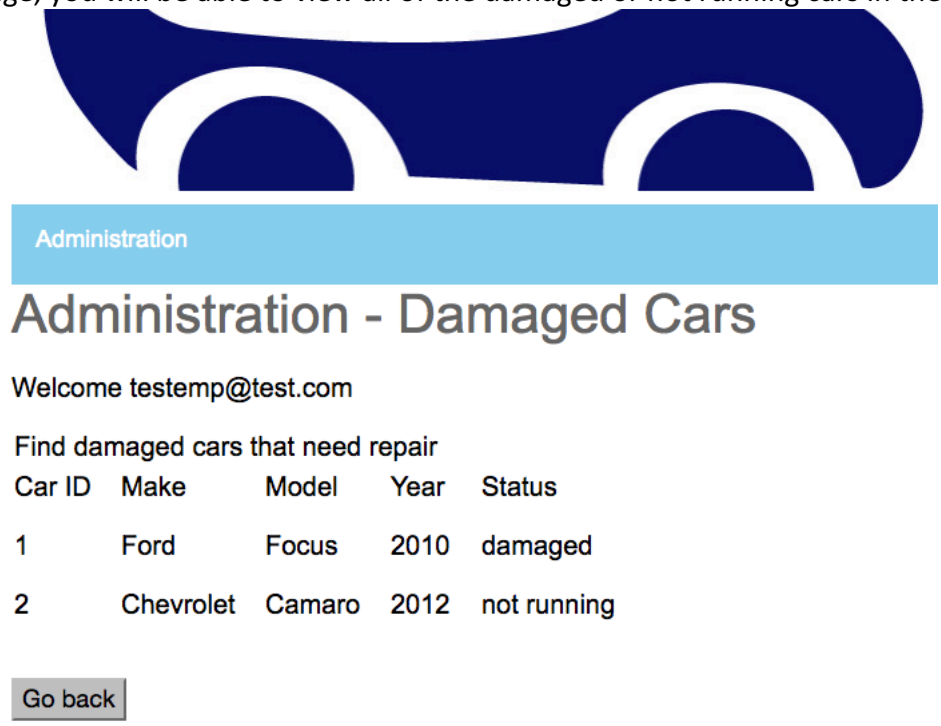
Go back

Over 5000km since Last Maintenance

On this page, you will be able to view all of the cars that have travelled over 5000kms since their last maintenance.

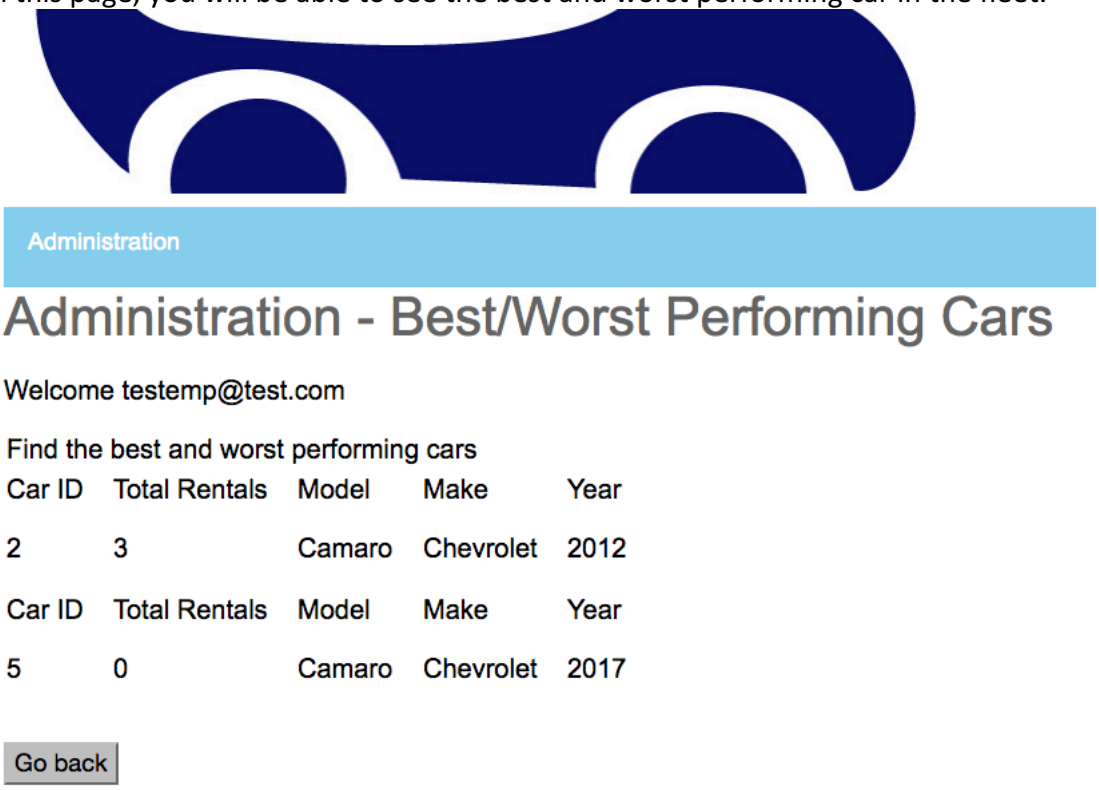
Damaged Cars

On this page, you will be able to view all of the damaged or not running cars in the fleet.



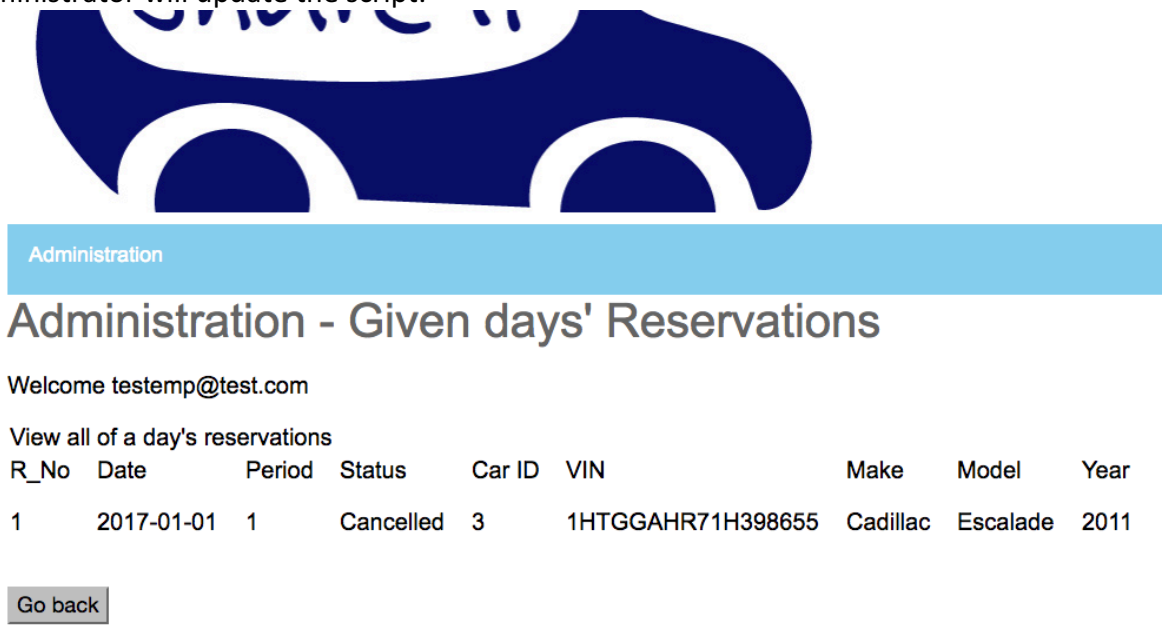
Highest/Lowest Rentals

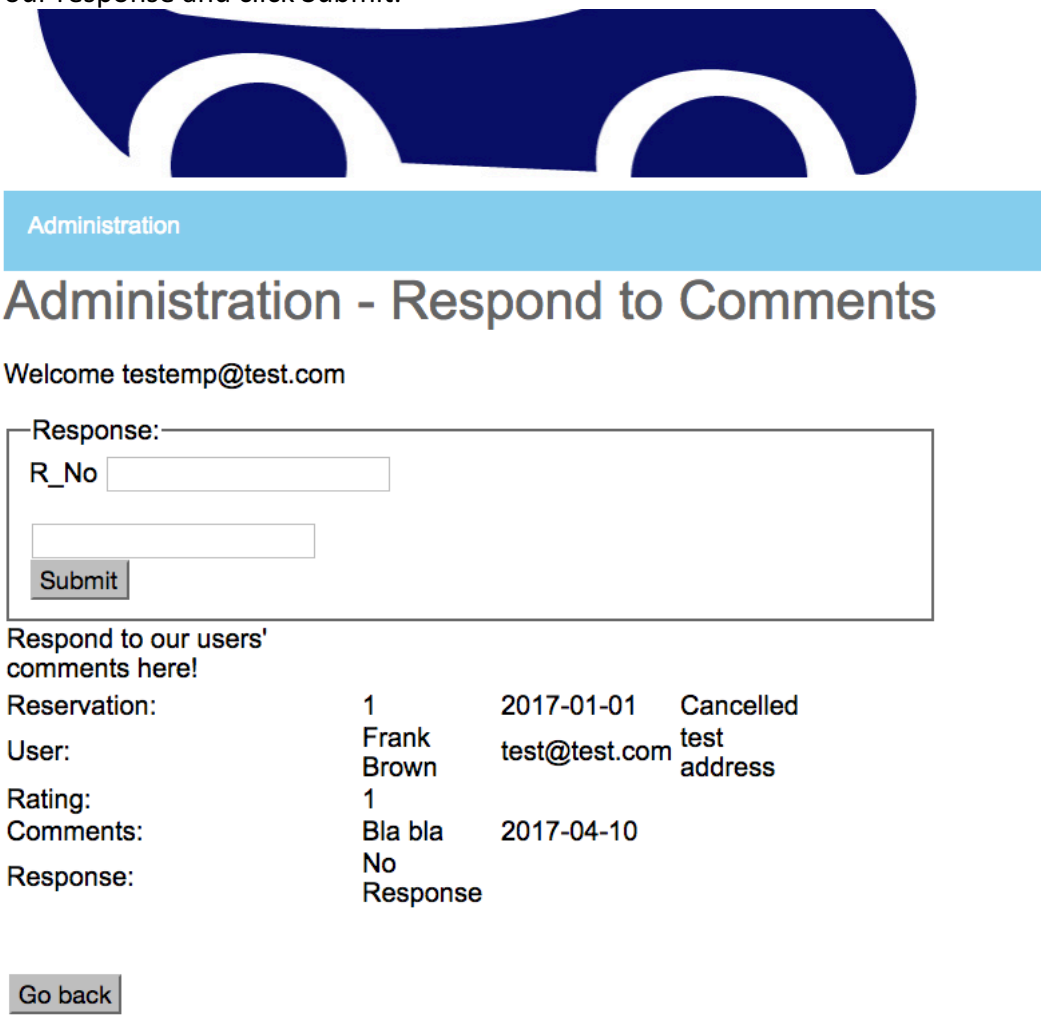
On this page, you will be able to see the best and worst performing car in the fleet.



All reservations for Given Day

On this page, you will be able to view all of today's reservations. On each given day, the administrator will update the script.



Respond to Comments

On this page, you will be able to see all comments that do not have an employee response yet. Type in the reservation number (R_No) for the comment you wish to respond to, as well as your response and click Submit.