

Project 2

United Auto Sales Web Application

Due: April 25, 2021 at 11:55PM

Group Assignment

Note: Project 2 is a group project and groups should consist of at least 2 persons and no more than 5 persons.

If you have not already done so, please ensure that you let me know by email (yannick.lynfatt@uwimona.edu.jm) who your group members.

Background

For this project we will create a fictional Auto Sales site called United Auto Sales that will allow users to add cars that they want to sell, search for available cars and be able to view some more information on those cars. Users can also add a car to their favourites so they can quickly get back to it.

Database Schema

Cars

Column Name	Data Type
id	Integer
description	String
make	String
model	String

Column Name	Data Type
colour	String
year	String
transmission	String
car_type	String
price	Decimal/Float
photo	String
user_id	Integer

Favourites

Column Name	Data Type
id	Integer
car_id	Integer
user_id	Integer

Users

Column Name	Data Type
id	Integer
username	String
password	String
name	String
email	String
location	String
biography	String
photo	String
date_joined	DateTime

Note: Ensure that you define the appropriate models and create your database migrations.

Key Functionality

You should be able to register for an account on our United Auto Sales web application. Once a user has an account, they should be able to login and see available cars. They can also search by the make or model of the cars. A user also has the ability to submit information for a car being sold as well as upload a photo. Users can also view the full details of a car and also "Favourite" cars that they like. And lastly a user can view their own profile and see the cars that they have added to their favourites.

Part 1

The aim of the first part of the project is to build the API for your United Auto Sales application. This includes creating routes (endpoints) for the user registration and login system as well as the ability to add and view cars, search for cars and favourite cars. See *Table 1* and accompanying note.

The API routes (endpoints)

Table 1: API Routes (endpoints)

HTTP Method	Route	Description
POST	/api/register	Accepts user information and saves it to the database
POST	/api/auth/login	Accepts login credentials as username and password
POST	/api/auth/logout	Logout a user
GET	/api/cars	Return all cars
POST	/api/cars	Used for adding new cars

HTTP Method	Route	Description
GET	/api/cars/{car_id}	Get Details of a specific car
POST	/api/cars/{car_id}/favourite	Add car to Favourites for logged in user
GET	/api/search	Search for cars by make or model
GET	/api/users/{user_id}	Get Details of a user
GET	/api/users/{user_id}/favourites	Get cars that a user has favourited.

Note: More details about the API and examples of the expected JSON responses can be viewed at: info3180project2api2021.docs.apiary.io.

Test to ensure that your API works by using either the Postman REST Client (<http://getpostman.com/>) or the Curl command line tool to make requests to your API routes (endpoints).

Part 2

You are required to use VueJS to build the front end of your web application that will interact with the API you built in Part 1. You should also ensure the following is in place:

1. You should use the VueRouter library to create routing for your frontend and implement some Vue components to represent the different pages. *See Table 2 below for a list of routes.*
2. A User should be able to Register for an account and Login to the website. *See Figure 2 and 3.*
3. You should generate and send the appropriate Authorization header with each request to your API (except the login and register API

routes) using a JWT e.g. **Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjMONTY3ODkwIiwibmFtZSI6IkpvaG4gRG91IiwiaWF0IjOnRydWV9.TJVA950rM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ**

4. When a user successfully logs in they should see the last 3 cars that have been added from all users. They should also see a search box, where they can search by the make or model of a car. *See Figure 4.*
5. A user can click on the "View more details" button/link for a car to see the full details on the car. The "Email Owner" button does not need to do anything for this project but should still be included on the page. A user should however be able to click on the "heart" icon to add the car to their favourites. *See Figure 5.*
6. A user can view their own profile by clicking 'My Profile' from the menu. *See Figure 6.* When viewing a users' profile, you should be able to see the details about the user as well as any cars they have favourited.
7. The user should be able to choose to add a new car for sale to the system at which point they will supply all the relevant details for the car and upload an image. *See Figure 7.*
8. You should display a success message if the user successfully adds a new car or failure otherwise.

Frontend Routes

Table 2: Frontend Routes

Route	Description
/	Display the homepage of the web application.
/register	Accepts user information and saves it to the database

Route	Description
/login	Accepts login credentials as username and password
/logout	Logout a user
/explore	View/Explore all posts by all users
/users/{user_id}	View user profile info as well as all Posts by that user
/cars/new	Allow the user to add a new post
/cars/{card_id}	View all the details about a car

Deploy the application to Heroku

The finished application should be deployed to Heroku. If you haven't already done so, ensure that you sign up for an account on the Heroku website (<https://heroku.com>) and then do the following.

Note: You will also need the Heroku CLI. See instructions at <https://devcenter.heroku.com/articles/heroku-cli>.

```
heroku login
heroku apps:create
git push heroku master
```

You will also need to ensure that you have a Database setup on Heroku for your application. To create a provision the Postgres database add-on and create a PostgreSQL database on Heroku, follow the instructions at the following link:

<https://devcenter.heroku.com/articles/heroku-postgresql#provisioning-the-add-on>

Note: You will be using the **hobby-dev** plan when creating your database.

```
heroku addons:create heroku-postgresql:hobby-dev
heroku config -s
```

You should then see something like the following:

```
postgres://
yecsapnzl1ttgcb:8860d3512549e3ebba04aa68ede74496e30041ff
458ea1d46d7c4ed7f5402af0@ec2-54-221-244-196.compute-1.a
mazonaws.com:5432/d9d6gbbbcjoc71g
```

This is the URI that you will use in your **SQLALCHEMY_DATABASE_URI** config option in your Flask Application. Ensure that you change driver (at the start of the URI) from **postgres** to **postgresql** for Flask SQLAlchemy.

You will also need to ensure you make a modification to your Heroku **Procfile**. This will allow you to run the migration you created earlier (and any other future migrations) to create/update your Heroku Database. The updated **Procfile** will look similar to this:

```
release: python manage.py db upgrade --directory migrations
web: gunicorn -w 4 -b "0.0.0.0:$PORT" app:app
```

Submission

Submit your code via the "Project 2 Submission" link on OurVLE. You should submit the following links:

1. Your Github repository URL for your Flask app e.g. <https://github.com/{yourusername}/info3180-project2>
2. Your URL for your Heroku app e.g. <https://{yourappname}.herokuapp.com>

Appendix

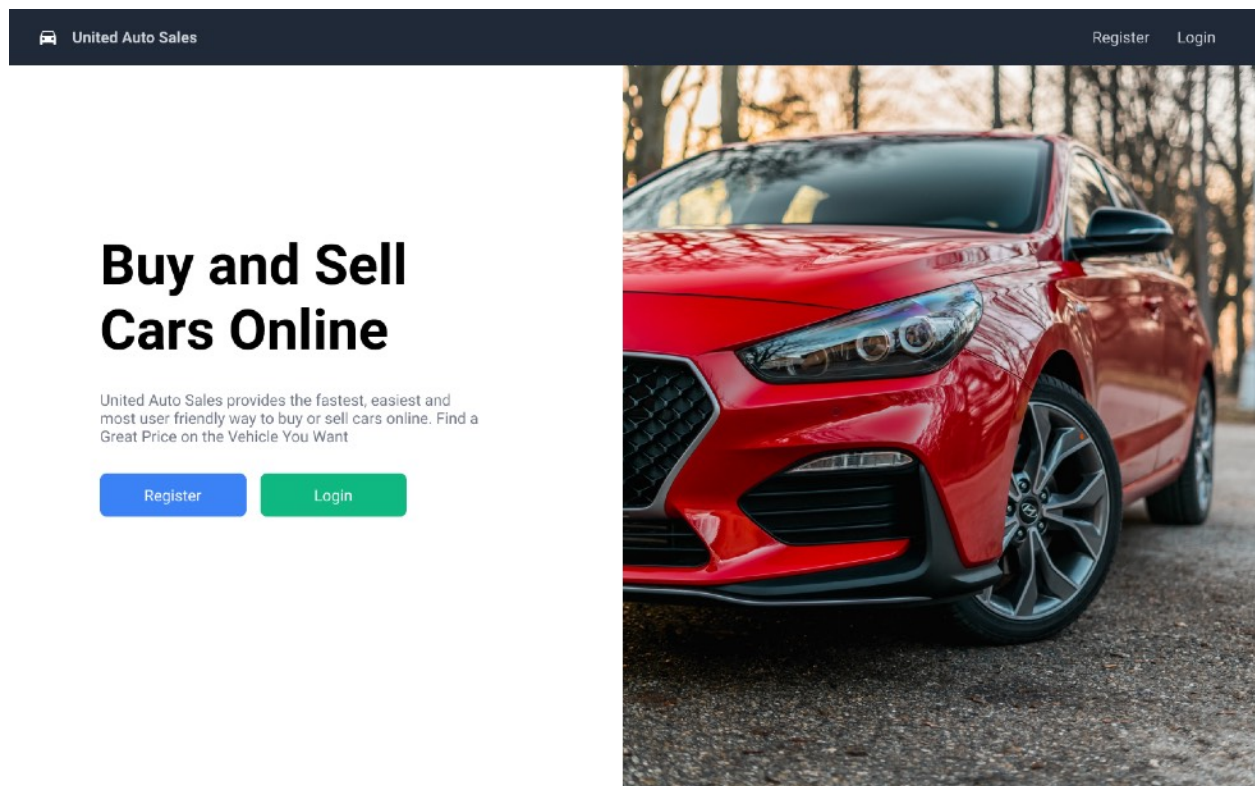



FIGURE 1. HOME PAGE

 United Auto Sales

RegisterLogin

Register New User

Username

Fullname

Location

Biography

Password

Email

Upload Photo

Browse

No File Selected

Register

FIGURE 2. USER REGISTRATION

United Auto Sales

RegisterLogin

Login to your account

Username

Password

Login

FIGURE 3. LOGIN

United Auto Sales

Add CarExploreMy Profile


Logout

Explore

Make

Model


Search



2020 Lamborghini Huracan

\$356,335


View more details



2020 Bugatti Chiron

\$3,062,888

View more details



2018 Tesla Model S

\$62,888

View more details

FIGURE 4: EXPLORE PAGE. THIS DISPLAYS CARS BY ALL USERS AND SEARCH

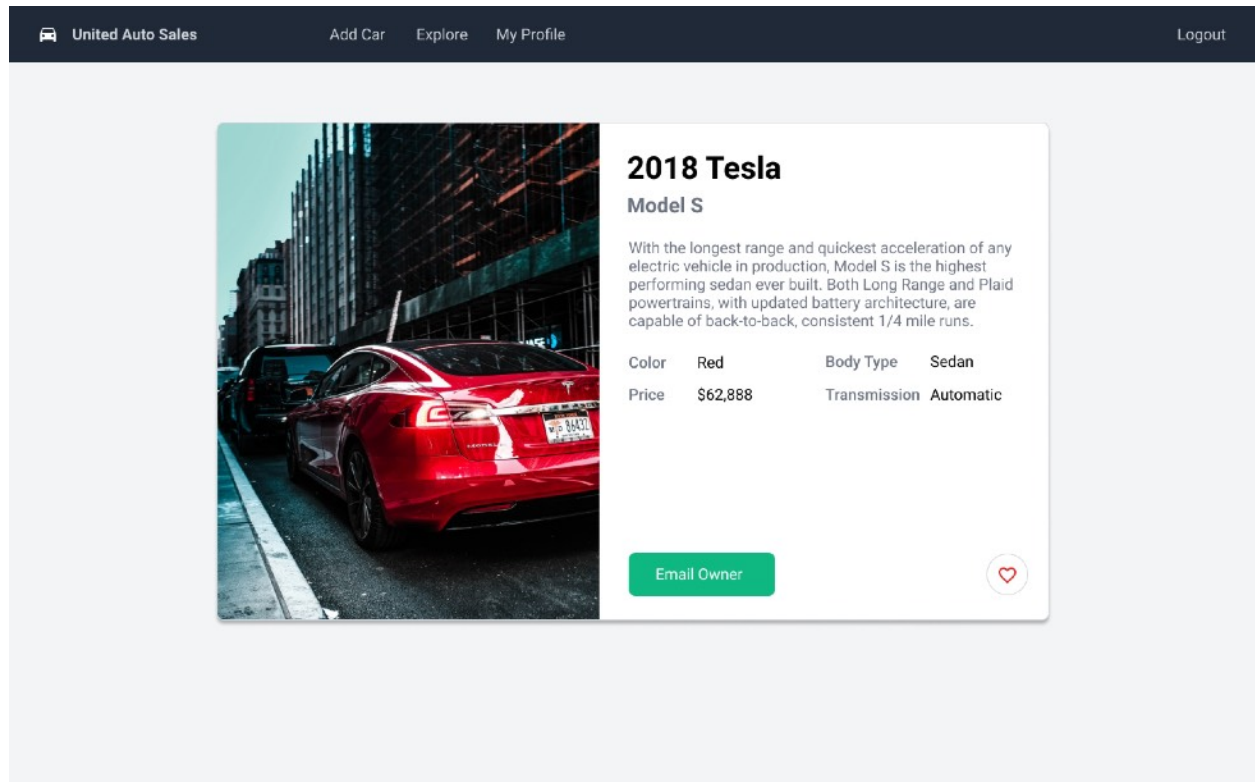
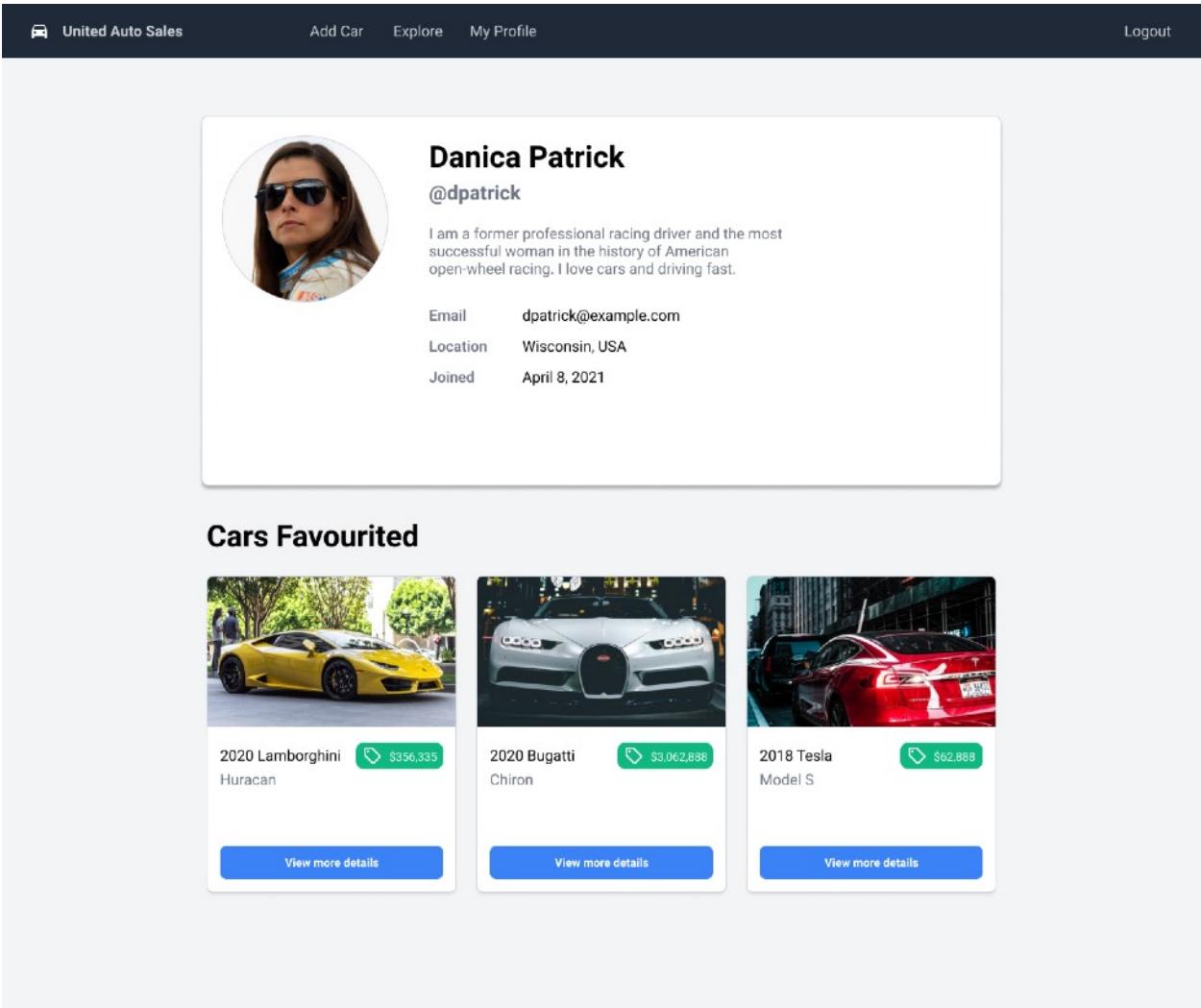



FIGURE 5: CAR DETAILS PAGE - WHEN THE HEART IS CLICKED TO FAVOURITE A CAR IT SHOULD CHANGE COLOUR.



 United Auto Sales

[Add Car](#) [Explore](#) [My Profile](#)

[Logout](#)

Add New Car

Make

Tesla

Model

Model S

Colour

Red

Year

2018

Price

62888

Car Type

SUV

Transmission

Automatic

Description

Upload Photo

Browse

No File Selected

Save

FIGURE 7: FORM FOR ADDING A NEW CAR