

A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with subtle diagonal lines.

Myriad Cars

By: Sam Babayan



About Myself

- I'm from Boston Massachusetts
- Always loved technology and taught myself to take apart/fix phones and build computers
- Originally Graduated MCPHS University with a Bachelors of Science in Pre-medical Studies
- Had an interest in cars since I was 15 and went to many auto shows
- Worked as an Auto Body Technician/Mechanic to save up money



Myraid Cars Functionality

- Links APIs(B2B)
- Displays available car inventory with details about each car
- Gives users ability to register/login
- Logged in users can add cars to their cart
- Admin Page gives admin users the ability to modify car and user tables
- Contact table has all of the Myraid Cars contact links

Wireframes

Logo

Register

First Name

Last Name

Email

Password

Register

Log in link

This wireframe shows a registration form centered on a light gray background. At the top left is a 'Logo' placeholder. The form is enclosed in a rounded rectangle and contains fields for 'First Name', 'Last Name', 'Email', and 'Password'. Below these fields is a blue 'Register' button and a 'Log in link'.

Logo

Log In

Email

Password

Log In

Register link

This wireframe shows a login form centered on a light gray background. At the top left is a 'Logo' placeholder. The form is enclosed in a rounded rectangle and contains fields for 'Email' and 'Password'. Below these fields is a blue 'Log In' button and a 'Register link'.

Logo

Log In

Fast Cars

Intro Text

View

Image

Website Features

Description

Contact Information

Column of media platform

Column of media account link

This wireframe shows a homepage layout. At the top are 'Logo' and 'Log In' placeholders. Below is a 'Fast Cars' header with 'Intro Text' and a 'View' button, followed by a placeholder image. The main content area is titled 'Website Features' and 'Description'. At the bottom, there is a 'Contact Information' section with two columns: 'Column of media platform' and 'Column of media account link'.

Logo

Log In

This wireframe shows a grid layout with six placeholder images arranged in two rows of three. A 'Logo' placeholder is at the top left and a 'Log In' placeholder is at the top right. A central vertical element separates the two columns of images.

Logo

Add Car

Update/Delete

Update/Delete

Update/Delete

Update/Delete

Add Customer

Update/Delete

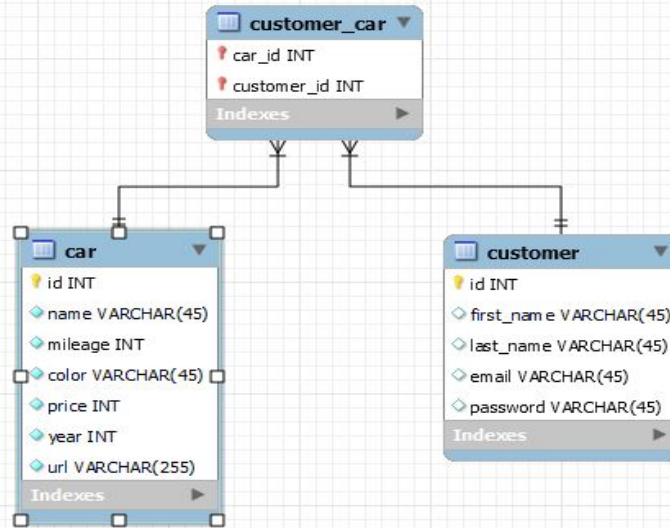
Update/Delete

Update/Delete

Update/Delete

This wireframe shows a form for adding a car. At the top are 'Logo' and 'Add Car' placeholders. Below is a form with four 'Update/Delete' buttons. Below this is a section for 'Add Customer' with four more 'Update/Delete' buttons.

Database Structure



Database Tables

id	name	mileage	color	price	year	url
1	Audi RS7	12114	Gray	100000	2021	https://images.dealer.com/ddc/vehicles/2022/A...
2	BMW M4	5438	Black	75000	2020	https://platform.cstatic-images.com/xlarge/in/v...
3	Range Rover SVR	34234	Blue	115000	2022	https://www.renderhub.com/3dstarving/range-...
4	Lamborghini Aventador	42743	Gold	320000	2017	https://www.lamborghini.com/sites/it-en/files/D...
15	Tesla Model X	65432	White	90000	2020	https://crdms.images.consumerreports.org/c_lfi...

Car table

Linked customer car table

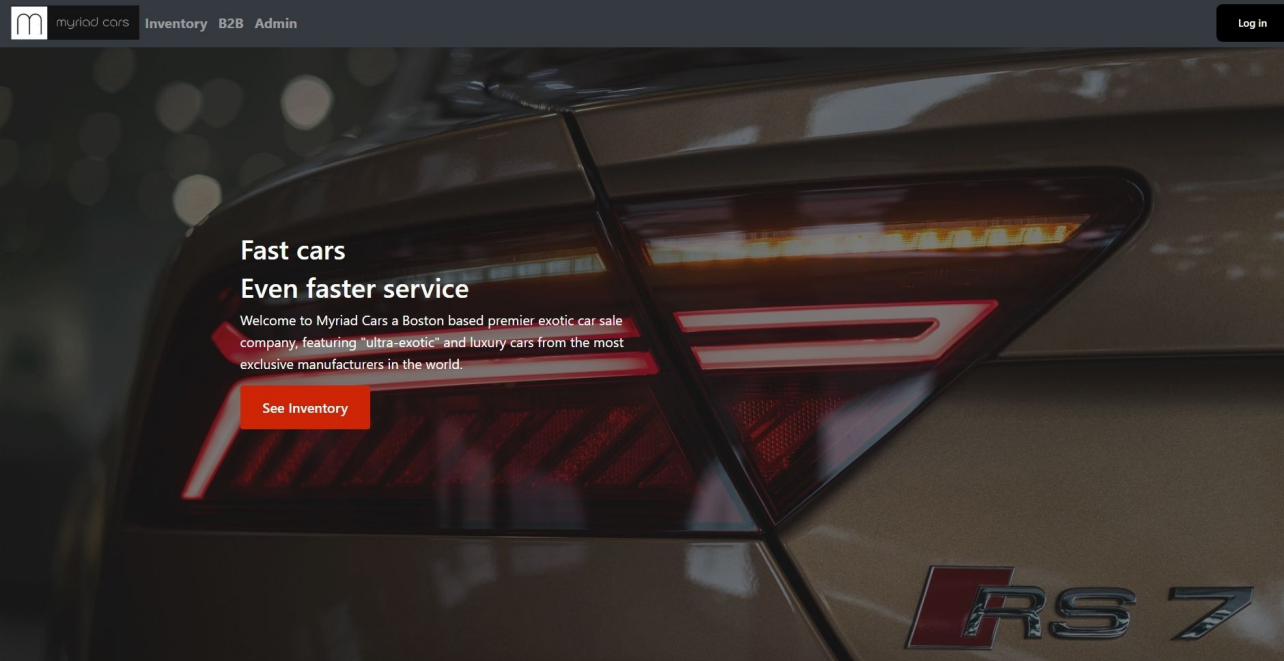
car_id	customer_id
1	34
1	46
3	46

Customer table

id	first_name	last_name	email	password
4	BOB	Public	Mary@mycapstone.com	a
5	Bob	Dixon	Bobwell@mycapstone.com	a
6	s	b	b	a
7	test	tttt	ttt	a
11	a	a	a	a
14	test1	test2	test3@gmail.com	a
22	c	cccccccccccccccc	c@gmail.com	a
23	Maria	Thomas	testg@gmail.com	a
34	a	a	a@a	a
44	z	z	z@z.com	z
46	jim	bob	jim@gmail.com	bob

Spring MVC, Hibernate, HTML, CSS, Bootstrap

Flow of Information: Request -> Controller
-> Service -> Dao -> Database



- ▼ Capstone-MVC
 - > Deployment Descriptor: Capstone-MVC
 - > JAX-WS Web Services
 - ▼ Java Resources
 - ▼ src
 - ▼ com.myraid.spring.controller
 - > CarController.java
 - > CustomerController.java
 - ▼ com.myraid.spring.dao
 - > CarDAO.java
 - > CarDAOImpl.java
 - > CustomerDAO.java
 - > CustomerDAOImpl.java
 - ▼ com.myraid.spring.entity
 - > Car.java
 - > Customer.java
 - ▼ com.myraid.spring.service
 - > CarService.java
 - > CarServiceImpl.java
 - > CustomerService.java
 - > CustomerServiceImpl.java
 - ▼ com.myraid.spring.testdb
 - > TestDbServlet.java
 - > Libraries
 - > Referenced Libraries
 - > build
 - > Capstone
 - > WebContent

JUnit Testing

```
1 package com.example.demo;
2
3 import org.junit.jupiter.api.Assertions;
4
5 @Transactional
6 @Rollback
7 @SpringBootTest(classes = MyraidJUnitTestApplication.class)
8 public class CustomerRepositoryTest {
9
10     @Autowired
11     CustomerRepository customerRepository;
12
13     @Test
14     void testCustomerRepository() {
15         Assertions.assertNotNull(customerRepository);
16     }
17
18     @Test
19     void findingCustomersByTheirId() {
20         Customer customer = new Customer();
21         customer.setFirstName("test");
22         customerRepository.save(customer);
23         Customer customer2 = customerRepository.findById(customer.getId()).get();
24         Assertions.assertNotNull(customer2);
25         Assertions.assertEquals("test", customer2.getFirstName());
26     }
27
28     @Test
29     void testDeleteingAUser() {
30         Customer customer = new Customer();
31         customer.setId(4);
32         customerRepository.save(customer);
33         customerRepository.deleteById(4);
34         Assertions.assertFalse(customerRepository.findById(4).isPresent());
35     }
36 }
37
38
39
40
41
42
43
44
45
46
47
```

Runs: 3/3

Errors: 0

Failures: 0

CustomerRepositoryTest [Runner: JUnit 5] (0.178 s)

findCustomerById() (0.146 s)
testDeleteUser() (0.027 s)
testCustomerRepository() (0.004 s)

Capstone-myraid-junit-test [boot] [devtools]

src/main/java

myraid.spring.test

MyraidJUnitTestApplication.java

myraid.spring.test.controller

CustomerController.java

myraid.spring.test.dao

CustomerRepository.java

myraid.spring.test.entity

Customer.java

myraid.spring.test.service

CustomerService.java

CustomerServiceImpl.java

src/main/resources

src/test/java

com.example.demo

CustomerRepositoryTest.java

MyraidJUnitTestApplicationTests.java

JRE System Library [JavaSE-1.8]

Maven Dependencies

JUnit 5

src

target

HELP.md

mvnw

mvnw.cmd

pom.xml







Spring Security, Thymeleaf (with Bcrypt)



myriad cars

Inventory B2B Admin

Add Car

Name	Mileage	Color	Price	Year	Url	Action
Audi R5S	12114	Gray	100000	2021		Update Delete
BMW M4	5438	Black	75000	2020		Update Delete
Range Rover SVR	34234	Blue	115000	2022		Update Delete
Aventador	42743	Gold	320000	2017		Update Delete
Tesla Model X	65432	White	90000	2020		Update Delete
Ferrari 458	3425	Red	264000	2018		Update Delete

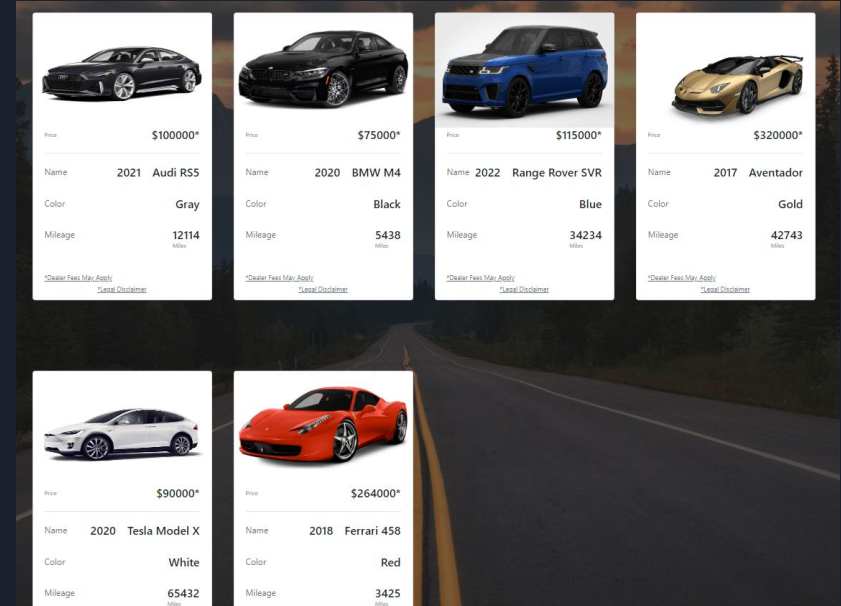
Add Customer

First Name	Last Name	Email	Action
BOB	Public	Mary@mycapstone.com	Update Delete
Bob	Dixon	Bobwell@mycapstone.com	Update Delete
test1	test2	test3@gmail.com	Update Delete
c	cccccccccccccccccc	c@gmail.com	Update Delete
Marla	Thomas	testg@gmail.com	Update Delete
Marla	Thomass	testg@gmail.commmmm	Update Delete
a	a	a@a	Update Delete
z	z	z@z.com	Update Delete

```
Capstone-myraid-security [boot] [devtools]
└─ src/main/java
    └─ com.myraid.springboot.thymeleafdemo
        ├── ServletInitializer.java
        ├── ThymeleafdemoApplication.java
        └─ com.myraid.springboot.thymeleafdemo.controller
            ├── CustomerController.java
            ├── DashboardController.java
            ├── DemoController.java
            └── LoginController.java
    └─ com.myraid.springboot.thymeleafdemo.dao
        └─ CustomerRepository.java
    └─ com.myraid.springboot.thymeleafdemo.entity
        └─ Customer.java
    └─ com.myraid.springboot.thymeleafdemo.security
        └─ SecurityConfiguration.java
    └─ com.myraid.springboot.thymeleafdemo.service
        ├── CustomerService.java
        └── CustomerServiceImpl.java
    └─ src/main/resources
    └─ src/test/java
    └─ JRE System Library [JavaSE-1.8]
    └─ Maven Dependencies
    └─ src
    └─ target
        ├── mvnw
        ├── mvnw.cmd
        └─ pom.xml
```

Car Inventory Page

- When users first access the website they are able to see lots of information about the company, with the main point being able to access the full list of cars available within our inventory



Login/Register

Register Your Account

First Name

Last Name

Email

Password

Register

Have already an account? [Login here](#)

- To gain further access within the website users must create an account for the website
- To log in users must provide a username and password that was previously registered within the website
- If the username and password match with the database information they are then forwarded to a new homepage with added functionality that only logged in users can access
- If the username and password do not match with the database they are redirected to the login page so they may attempt to login again.

Welcome to Myraid Cars

Email

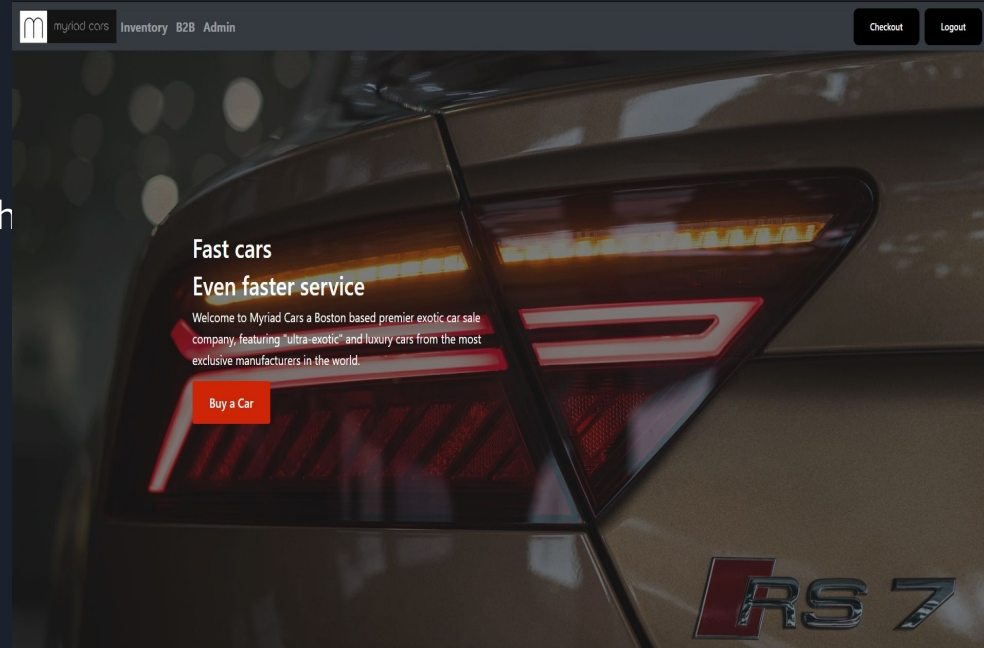
Password

Log In

Don't have an account? [Register Here](#)

Gaining Further Access

Once logged in, users will be met with a similar looking homepage however they might notice a few differences







Buying Cars

As the user is now logged in they are able to see a list of cars that are currently available with the ability to buy them. If they hit the “Buy” button on the selected car it will be added to their cart and taken away from the current inventory view.

Image	Price	Name	Color	Mileage	Action	Disclaimer
	\$100000*	2021 Audi R55	Gray	12114 Miles	Buy	*Legal Disclaimer
	\$75000*	2020 BMW M4	Black	5438 Miles	Buy	*Legal Disclaimer
	\$115000*	2022 Range Rover SVR	Blue	34234 Miles	Buy	*Legal Disclaimer
	\$90000*	2020 Tesla Model X	White	65432 Miles	Buy	*Legal Disclaimer
	\$264000*	2018 Ferrari 458				

Cart System

- The logged in user is able to access their cart which will display the cars they have added.
- The user has the ability to delete selected cars from their cart, which will then add the car back to the inventory page if they wish to add it back to the cart.

Image	Name	Year	Mileage	Color	Price	
	Audi RS7	2021	Gray	12114	100000	Delete
	Range Rover SVR	2022	Blue	34234	115000	Delete
	Lamborghini Aventador	2017	Gold	42743	320000	Delete
	Tesla Model X	2020	White	65432	90000	Delete



Potential Future Applications

- Use as an actual car sales website
- Add functionality for users to list their own cars for sale
 - Develop a mobile application for a more fluid experience
- Sell to dealerships looking to modernize their website



Challenges

- Had to create a separate project for JUnit tests because my spring security project ran on an older version of spring security which caused issues when trying to incorporate Junit within the project
- Cart system was tricky to create and took many hours of debugging in order to get working correctly
- Incorporating an external CSS script can sometimes require you to manually empty the web browser cache and hard reload the website in order to have any new changes be applied



Conclusion

- I learned a lot through my experience of creating this capstone project and while at times it was strenuous, overall it was an enjoyable experience
- I am a hands on learner so putting myself through the process of starting with an idea and trying to bring it to life helped me understand the responsibilities and challenges that come with being a full stack developer