CEN3024C Module 5 Project

For this project we will be implementing a CORS policy at the global level and a "one-off" method-level policy to the PizzaRestaurant project. We will also be adding support for a password for the customers.

1. Add a global policy using a configuration class (add this in a new "config" package):

```
@Configuration
public class CorsConfig implements WebMvcConfigurer {
    @Override
    public void addCorsMappings(CorsRegistry registry) {
        registry.addMapping("/**")
                .allowedOrigins("http://example.com")
                .allowedMethods("GET", "POST", "PUT", "DELETE")
                .allowedHeaders("*")
                .allowCredentials(true);
    }
}
```

2. Choose an endpoint in your controllers and add a method-based policy:

```
@CrossOrigin(origins = "http://example2.com",
        methods = {RequestMethod.GET, RequestMethod.POST},
        allowedHeaders = "*", allowCredentials = "true")
```

I added mine to the @GetMapping("/{pizzatype}") endpoint in the PizzaController class. Technically it doesn't matter if the @CrossOrigin annotation preceeds or follows the @GetMapping annotation, but convention dictates that it should go before @GetMapping.

Test and verify the correct operation of your policies with curl by specifying the Origin field.

- Any request using the **example.com** origin should return a 200 and display the corresponding data, e.g.,

```
curl -i -H "Origin: http://example.com" -X GET http://localhost:8080/api/customers

HTTP/1.1 200
...
[{"lastName":"Brown","firstName":"Sally","address":"2 Pine Ln.","phoneNumber":"407-555-
2323","userId":"sallybrown","email":null},{"lastName":"Smith","firstName":"John","address":
"1 Oak St.","phoneNumber":"904-555-1212","userId":"johnsmith","email":null}]
```

- Any request using the **example2.com** origin should return a 403 and display an "Invalid CORS request" error, except for the request where you have added the method-based policy, e.g.,

```
curl -i -H "Origin: http://example2.com" -X GET http://localhost:8080/pizzas/pepperoni

HTTP/1.1 403
```

```
...
Invalid CORS request
```

3. Add a salt and hash field to the customer entry and a password field to the customer DTO. Use the code examples provided in the lecture slides to provide code that creates a new user using the userId field (not the customerId) as the identifier. Verify the userId does not exist before adding the user.

Submit your solution to the GitHub classroom assignment repo.