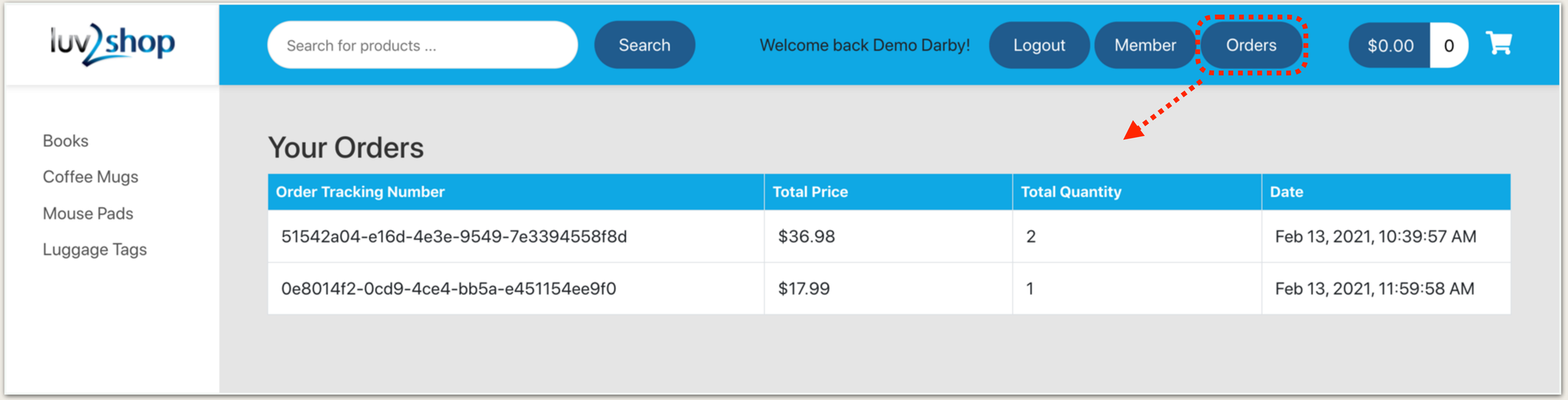
View Order History

View Order History

- Allow users to view their previous order history
- This feature is only available to logged in / authenticated users



Front End and Back End

- Split up the development across multiple videos
- We'll start working on Spring Boot back end
- Then we'll work on Angular front end

Development Process - Spring Boot

1. Create OrderRepository REST API
2. Update configs to make OrderRepository REST API read only

Step 1: Create OrderRepository REST API

- Expose endpoint to search for orders by the customer's email address

HTTP Method	Endpoint
`GET`	`/api/orders/search/findByCustomerEmail?email=demo@luv2code.com`

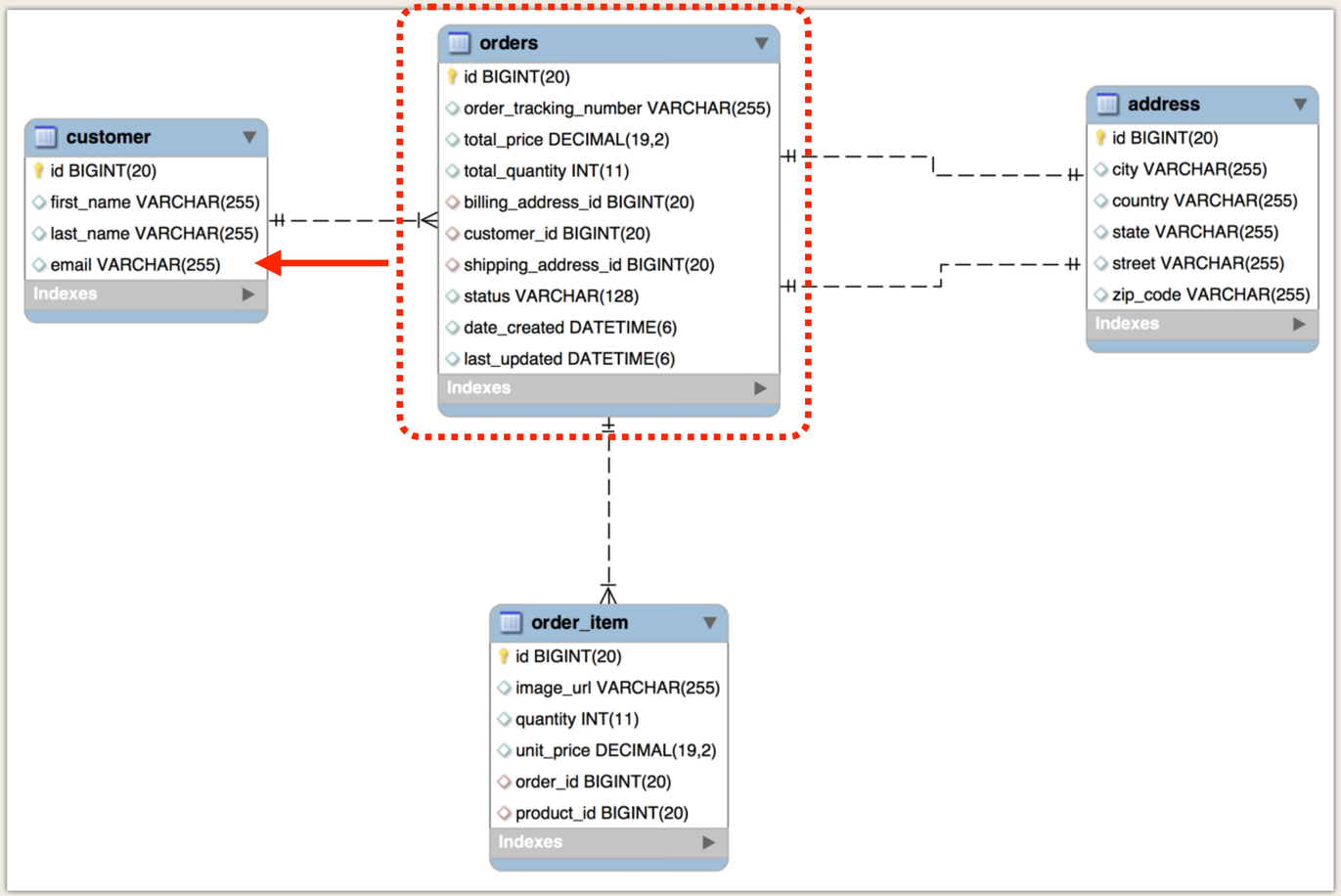
Step 1: Create OrderRepository REST API

- Spring Data REST and Spring Data JPA supports "query methods"
- Spring will construct a query based on method naming conventions

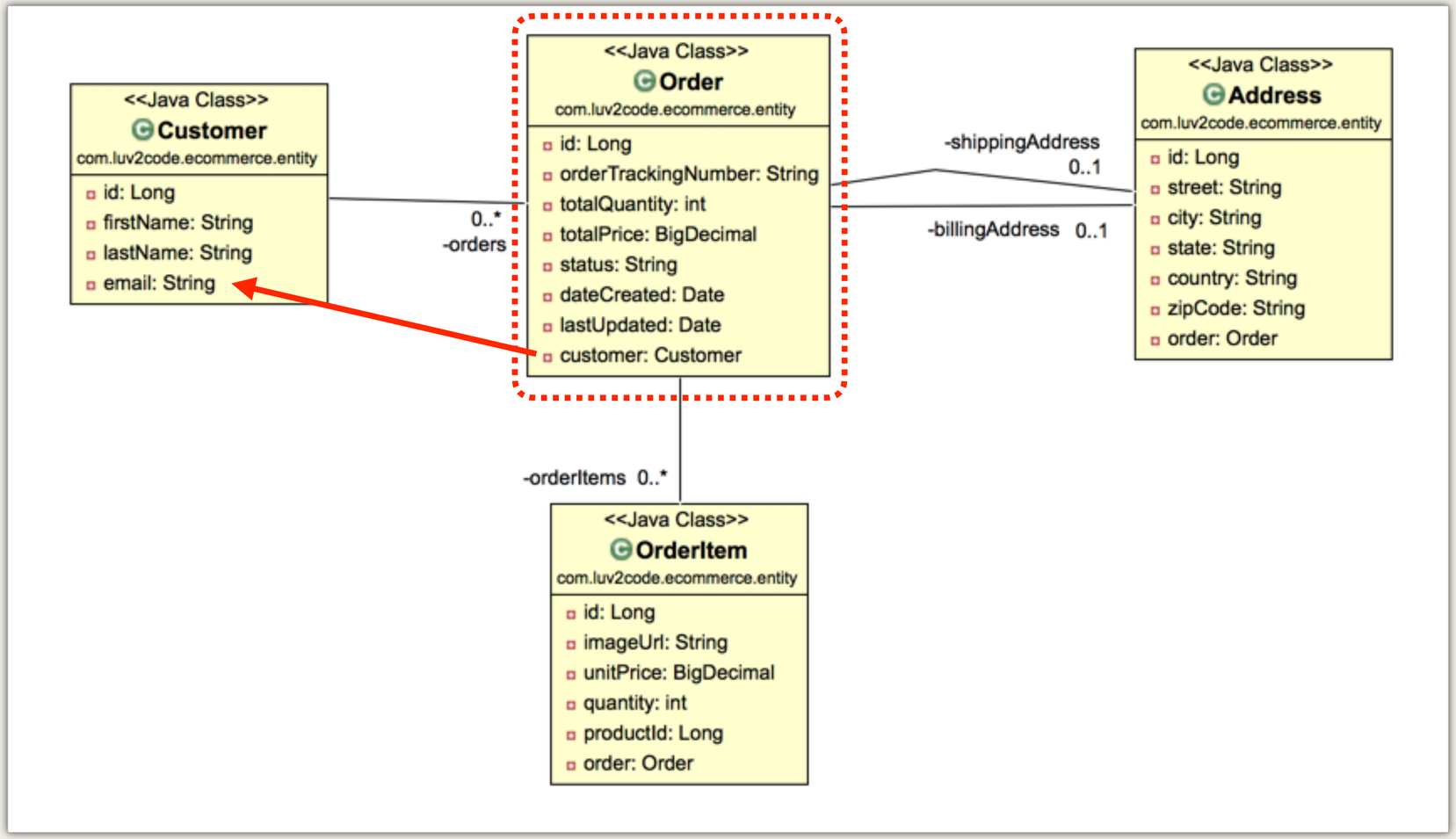
File: OrderRepository.java

```
@RepositoryRestResource
public interface OrderRepository extends JpaRepository<Order, Long> {

    Page<Order> findByCustomerEmail(@Param("email") String email, Pageable pageable);

}
```

Database Diagram



Class Diagram - Entity Classes



Step 1: Create OrderRepository REST API

File: OrderRepository.java

```
@RepositoryRestResource
public interface OrderRepository extends JpaRepository<Order, Long> {

    Page<Order> findByCustomerEmail(@Param("email") String email, Pageable pageable);

}
```

Behind the scenes, Spring will execute a query similar to this

```
SELECT * FROM orders
LEFT OUTER JOIN customer
ON orders.customer_id=customer.id
WHERE customer.email=:email
```

Step 1: Create OrderRepository REST API

File: OrderRepository.java

```
@RepositoryRestResource
public interface OrderRepository extends JpaRepository<Order, Long> {

    Page<Order> findByCustomerEmail(@Param("email") String email, Pageable pageable);

}
```

http://localhost:8080/api/orders/search/findByCustomerEmail?email=demo@luv2code.com

To pass data to REST API

Step 2: Update configs to make OrderRepository REST API read only

File: MyDataRestConfig.java

```
@Configuration
public class MyDataRestConfig implements RepositoryRestConfigurer {

 @Override
 public void configureRepositoryRestConfiguration(RepositoryRestConfiguration config, CorsRegistry cors) {

    HttpMethod[] theUnsupportedActions = {HttpMethod.PUT, HttpMethod.POST,
                                          HttpMethod.DELETE, HttpMethod.PATCH};

     // disable HTTP methods for ProductCategory: PUT, POST and DELETE
    disableHttpMethods(Product.class, config, theUnsupportedActions);
    disableHttpMethods(ProductCategory.class, config, theUnsupportedActions);
    ...
    disableHttpMethods(Order.class, config, theUnsupportedActions);

  }

  ...
}
```