Learning Consolidation

Manage Semi-structured and Unstructured Data and Handle Exceptions Within a RESTful Service by Using Mongo Repository

Learning Objectives

- Create a RESTful API using Spring Data MongoDB and Spring Boot.

Slide Note

Returns the entity identified by the given ID

MongoRepository

- `MongoRepository` is an interface provided by Spring Data to work with MongoDB.
- It extends `CrudRepository`.
- It provides all the necessary methods that help create a CRUD application and it also supports the custom-derived query methods.
- Some CRUD methods:
  - `S save(S entity)` - Saves the given entity.
  - `findById(ID primaryKey)` - Returns the entity identified by the given ID.
  - `S insert(S entity)` - Inserts the given entity.

Click on the link to refer to docs.

Slide Note

Repository Layer

- Spring Data `MongoRepository` helps with accessing the data in the MongoDB database.
- The `MongoRepository` provides common functionalities that we can easily plug in and use.
- Customer Repository extends the `MongoRepository`.
- The `Customer` is the domain class, which is a document, and the `Integer` is the dataType of the objectId.

```
public interface CustomerRepository extends MongoRepository<Customer,Integer> {

}
```

Document

Data type of the Id

Slide Note

Any data we wish to retrieve based on any other attribute other than the attribute marked with @Id from the database can be retrieved by using the findByAttribute() method. Here, attributes are the attributes of @Document class.

Repository Layer (contd.)

- Any data we wish to retrieve, based on any other attribute other than the attribute marked with `@Id` from the database can be retrieved by using the `findByAttribute()` method. Here, attributes are the attributes of the @Document class.
- You can only write these methods for the domain class that is annotated with `@Document`.
- The implementing class will be given by the Spring Data.
- Follow the conventions given by Spring Data to write these methods, and the implementation will be taken care of by the Spring Data.

```java
public interface CustomerRepository extends MongoRepository<Customer,Integer> {
    int findByCustomerPhoneNo(long customerPhoneNo);
    String findByCustomerName(String customerName);
}
```

Slide Note

Instead, you can write a custom query by using !

Repository Layer (contd.)

- Consider a scenario where we want to get information about all the customers who belong to a particular city.
- In this scenario, the city is not the attribute of the document `Customer`, so we cannot use `findByX()` methods of `MongoRepository`.
- Instead, you can write a custom query by using the `@Query` annotation as `Address` is an attribute of the `Customer` class.

```
public interface CustomerRepository extends MongoRepository<Customer,Integer> {
    @Query("{'customerAddress.city' : {$in : [?0]}}")
    List<Customer> findAllCustomerFromCity(String city);
}
```

The placeholder *?0* references the first parameter of the method.

This is the Address attribute in the Customer class.

Attribute of Address class

$in – here `in` is operator

- Click on link for docs.

Slide Note

Exception Handling

- Exception handling is an integral part of any application development process.
- The layers of a RESTful application must handle the exceptions appropriately:
  - `Service Layer`

    The Service layer performs the business logic of the application and communicates with the database; thus, all methods must declare exceptions.

  - `Controller Layer`

    A RESTful application can communicate the success or failure of an HTTP request by returning the right status code in response to the client.

    If the Service layer throws an exception due to a database failure, etc., the controller sends an error status code back to the client.

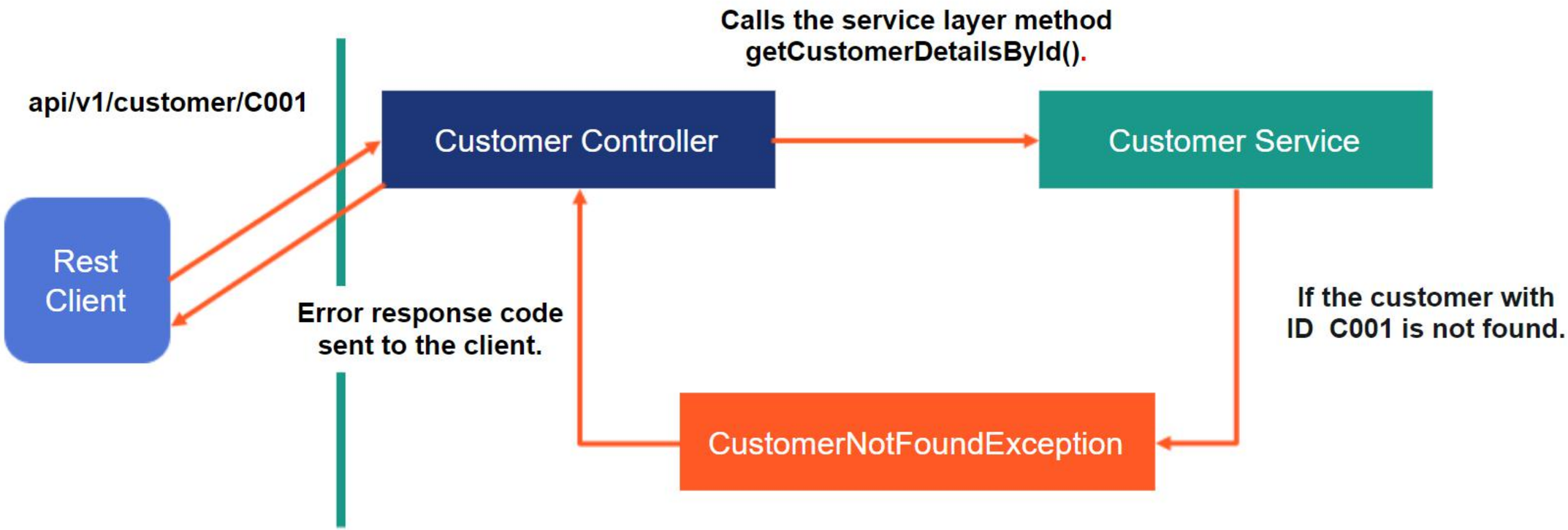
  - `Client side`

    An appropriate status code can help the client identify problems that might have occurred while the application was dealing with the request.

Slide Note

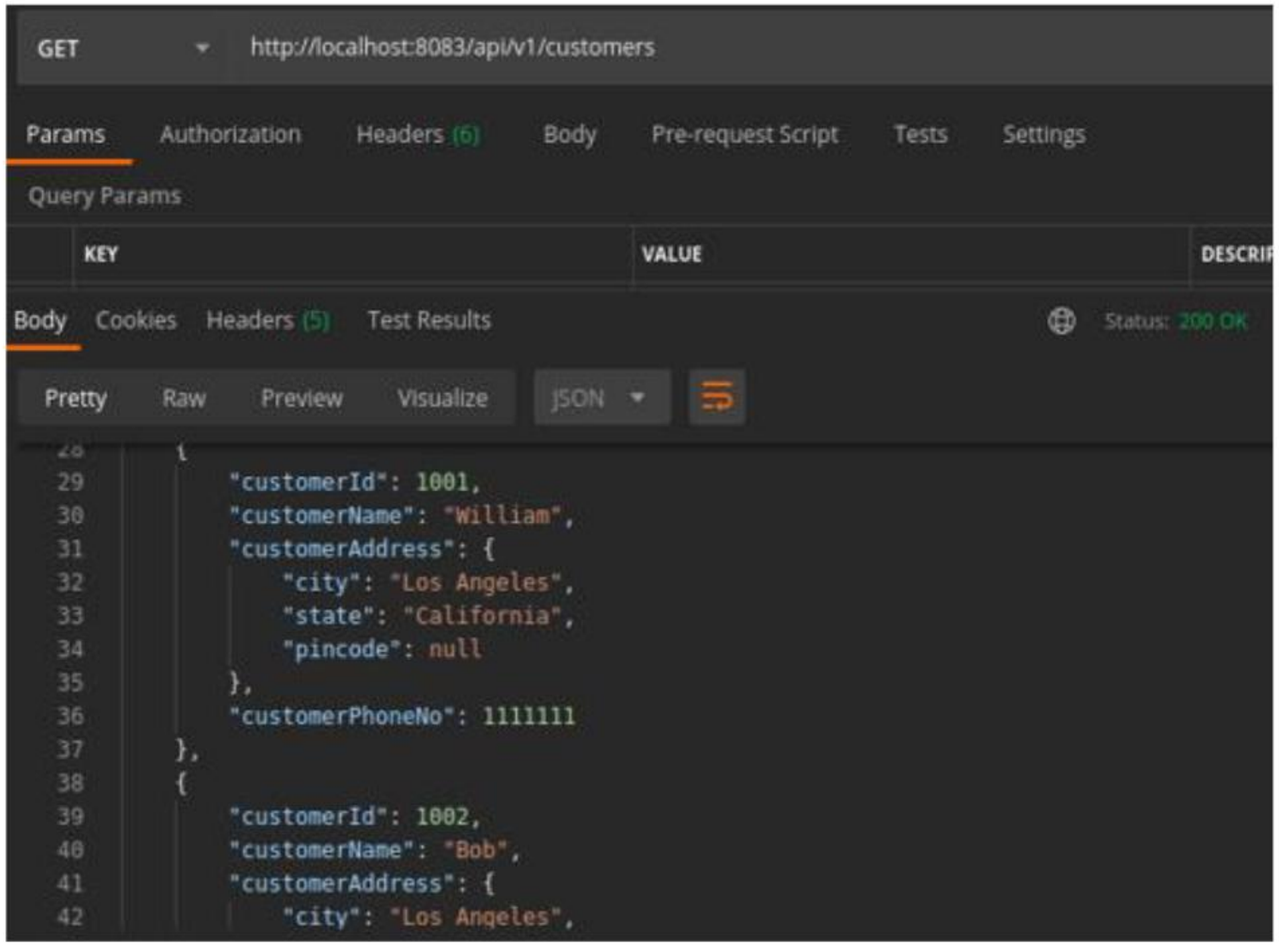
- From POSTMAN, the call is made to the appropriate handler.
- The handler then calls the service methods to process the business logic of the request. (In this case, get a product by the product code).
- If the product is not found, the error handling is done and a response is sent back to the client (POSTMAN).

Exception Propagation in RESTful Application



Slide Note

Postman Output



Slide Note

Postman Output With an Exception