Cycle:- Test Case 49

1). Entity class :- @Entity
@Table (name = "Orders")
Public Class Order {.
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)

② DTO class:-

i) Name Validator, 1st Letter should be capital followed by all Lowercase letters.

@NotNull (message = "{ name.notpresent ⇒ refer from msg prop.}")
@Pattern (regexp = "[A-Z][a-z]+(\\s[A-Z][a-Z]+)*",
message = "{ msg prop }")
invalid-name.

↗
Name

ii) Email:- valid email
@NotNull (message = "{ ---.notpresent}")
@Pattern (regexp= "[a-z 0-9] + [@] ([illegible] infy) (.com)",
∴ message = "{ --.invalid}")

iii) Repository:- Extend repository interface.

- The method should accept an emailId & return all orders of customer with emailId as List <Order>.
- The method should accept CycleType & return all orders details of particular cycleType as List <order>.

Ans. extends CrudRepository< Order , Integer>
|
Entity
Class
name

{

List< Order> findByEmailId (String emailid);

List <Order> findByCycleType (String Cycletype);

}.

iv) Validator :- a) Method rec. an object of OrderDTO.
& calls to validate the Value & throw excepti.

```
public static void validate (OrderDTO orderDTO) throws - ~
{ →
if (isValidDateOfOrder (orderDTO. get dateOfOrder()) = = false)
 {
   throw, new tourInfyCycleException(" msg from app.");
                                            prop
 }-
```

b) ~~&~~ • It validate dateOfOrder.
• Date should be at Least 14 days before today's date

• Above fails → false, else true.

```
  public static Boolean isValidDateOfOrder
Ans. if ( dateoforder .. isAfter (LocalDate. now()) ||
       date oforder. isBefore (LocalDate .now() . minusDays (14))
                                                            ;
  {
         return ~~false;~~ true;
     }  return ~~true;~~ false;
     }.
```