This document provides validation guidelines for key entities in the Panda online shopping platform as part of a business analysis exercise.

Website Name: Panda

URL: https://panda.sa/en

Panda is a Saudi online store that offers food and consumer products for online purchase. The website allows customers to browse products, add them to their cart, and complete the purchase process via electronic payment or upon receipt.

Selected Entities :

1. Customer
2. Product
3. Order

Entity 1: **Customer**

Validation Rule / Criteria: Must not be empty, minimum 3 characters
Consequence if Not Validated: Incomplete user profile; registration may fail

Code in Spring Boot:

```
@NotEmpty(message = "Name must not be empty")
@Size(min = 3, message = "Name must be at least 3 characters")
private String name;
```

Field: email

Validation Rule / Criteria: Must be a valid email format
Consequence if Not Validated: Customer may not receive confirmation or password reset emails

Code in Spring Boot:

```
@NotEmpty(message = "Email must not be empty")
@Email(message = "Invalid email format")
private String email;
```

Field: phone
Validation Rule / Criteria: Must start with 05 and contain exactly 10 digits
Consequence if Not Validated: No delivery or SMS updates will be sent

Code in Spring Boot:

```
@Pattern(regexp = "^05\\d{8}$", message = "Phone must start with 05
and be 10 digits")
private String phone;
```

Entity 2: **Product**

Field: name
Validation Rule / Criteria: Must not be empty
Consequence if Not Validated: Product display in the store will be broken or blank

Code in Spring Boot:

```
@NotEmpty(message = "Product name must not be empty")
private String name;
```

Field: price
Validation Rule / Criteria: Must be greater than 0
Consequence if Not Validated: Product may appear free or cause pricing errors

Code in Spring Boot:

```
@NotNull(message = "Price is required")
@Positive(message = "Price must be greater than zero")
private Double price;
```

Field: quantity
Validation Rule / Criteria: Must be 0 or more
Consequence if Not Validated: Customers may be allowed to order out-of-stock items

Code in Spring Boot:

```
@NotNull(message = "Quantity is required")
@Min(value = 0, message = "Quantity cannot be negative")
private Integer quantity;
```

Entity 3: **Order**

Field: orderId
Validation Rule / Criteria: Must not be empty and must be unique
Consequence if Not Validated: Orders cannot be tracked or may overlap

Code in Spring Boot:

```
@NotEmpty(message = "Order ID is required")
private String orderId;
```

Field: orderDate
Validation Rule / Criteria: Must be today or in the past
Consequence if Not Validated: System will show incorrect analytics and reporting

Code in Spring Boot:

```
@NotNull(message = "Order date is required")
@PastOrPresent(message = "Order date cannot be in the future")
private LocalDate orderDate;
```

Field: paymentMethod
Validation Rule / Criteria: Must be one of [cash, mada, visa]
Consequence if Not Validated: Invalid payment type may crash the order process

Code in Spring Boot:

```
@NotEmpty(message = "Payment method is required")
@Pattern(regexp = "^(cash|mada|visa)$", message = "Invalid payment
method")
private String paymentMethod;
```

These validation rules ensure data integrity and prevent potential issues during registration, ordering, and inventory management.