Answer the following

1. **What is the use of @valid Annotation**

Ans –

Use of @Valid Annotation:

The @Valid annotation is part of the Java Bean Validation framework (JSR 380) and is used to trigger validation on an object or its nested objects. It is typically applied in methods, especially in Spring MVC or Spring Boot applications, to ensure that the input data conforms to the specified validation rules before processing.

- **Primary use case**:
    - When applied to a method parameter, it ensures the parameter is validated.
    - When applied to a nested object, it validates the fields of the nested object.

Example –

```
public class User {
   @NotNull(message = "Name cannot be null")
   private String name;

   @Email(message = "Email should be valid")
   private String email;

   // getters and setters
}

@RestController
public class UserController {

   @PostMapping("/users")
```

```
    public ResponseEntity<String> createUser(@Valid @RequestBody User
user) {
        // Validation happens here. If invalid, throws
MethodArgumentNotValidException.
        return ResponseEntity.ok("User is valid");
    }
}
```

2. **Explain some predefined validator annotations.**

Ans –

Predefined Validator Annotations:

Here are some commonly used predefined validator annotations from **Jakarta Bean Validation** (formerly Hibernate Validator):

1. **@NotNull**
    - Ensures that the annotated field is not null.
    - Example: @NotNull(message = "Value cannot be null")
2. **@NotEmpty**
    - Ensures that the annotated field is not null or empty.
    - Applicable to collections, strings, or arrays.
    - Example: @NotEmpty(message = "List cannot be empty")
3. **@NotBlank**
    - Ensures that the annotated string is not null, empty, or contains only whitespace.
    - Example: @NotBlank(message = "Name cannot be blank")
4. **@Size**
    - Validates the size of collections, arrays, or strings.
    - Attributes: min and max.
    - Example: @Size(min = 3, max = 20, message = "Size must be between 3 and 20")

5. **@Min and @Max**
    - Validate numeric values to be at least a minimum or at most a maximum.
    - Example:

      @Min(value = 18, message = "Age must be at least 18")

      @Max(value = 60, message = "Age must not exceed 60")

      private int age;
6. **@Pattern**
    - Validates a string against a regular expression.
    - Example: @Pattern(regexp = "[A-Za-z]{3,}", message = "Must contain only letters")
7. **@Email**
    - Validates if the string is a well-formed email address.
    - Example: @Email(message = "Invalid email address")
8. **@Future and @Past**
    - Ensure dates are in the future or past.
    - Example:

      @Future(message = "Date must be in the future")

      private LocalDate deliveryDate;
9. **@Positive and @Negative**
    - Validates that a number is positive or negative.
    - Example: @Positive(message = "Value must be positive")
10. **@AssertTrue and @AssertFalse**
    - Ensure that the value of a boolean field is true or false.
    - Example: @AssertTrue(message = "The field must be true")