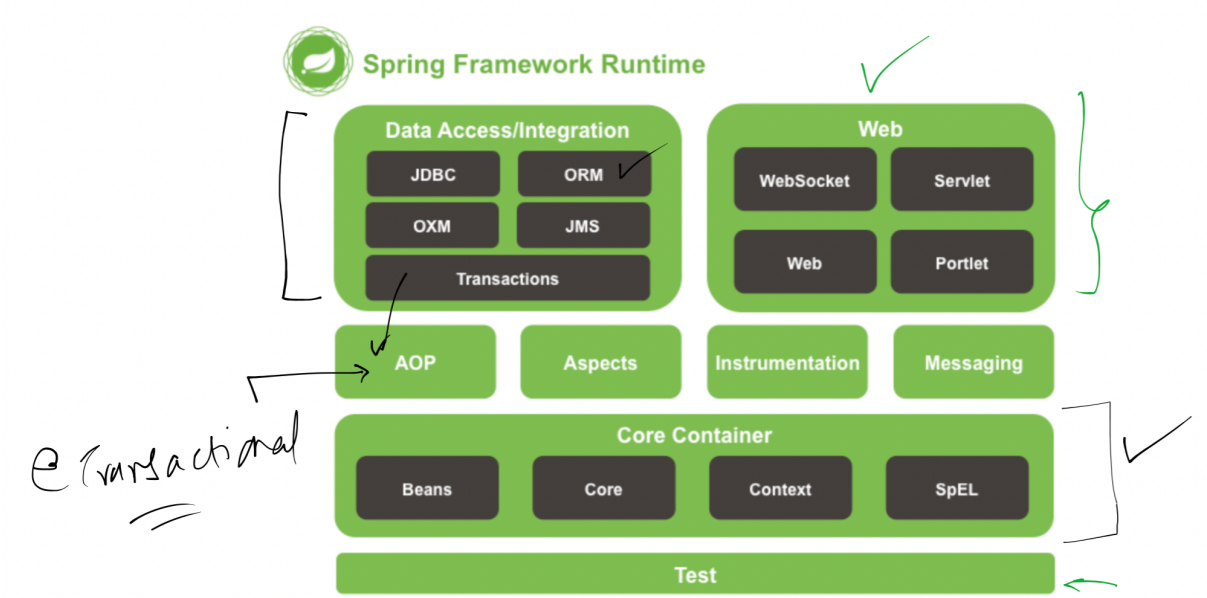
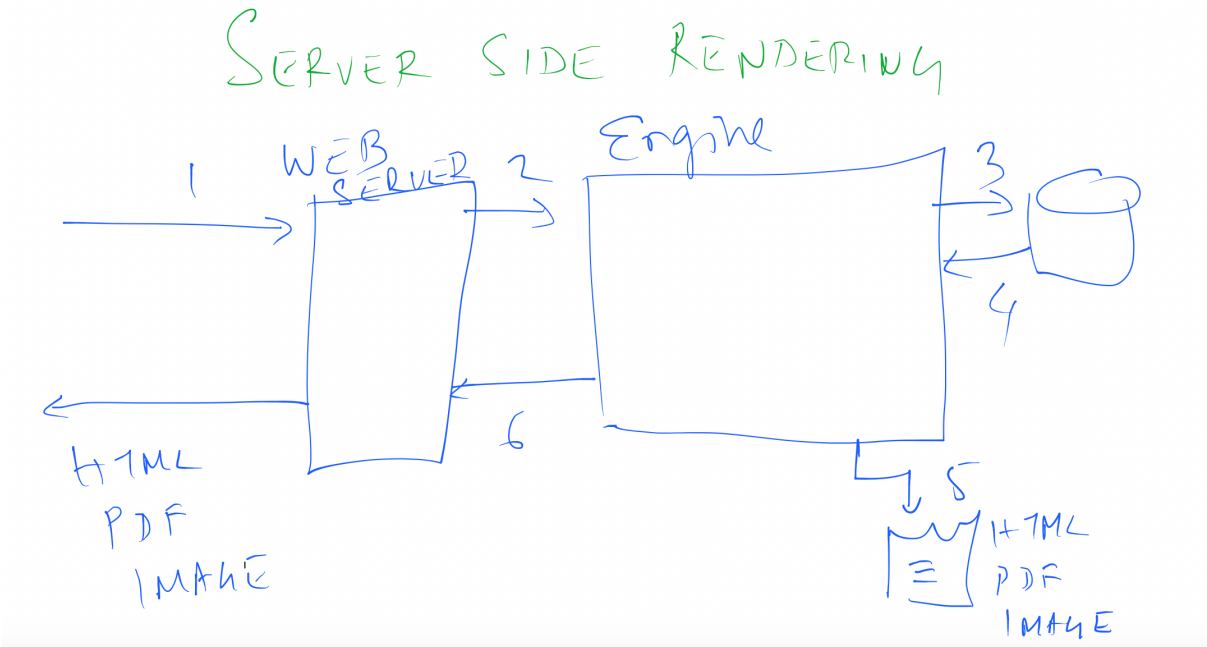
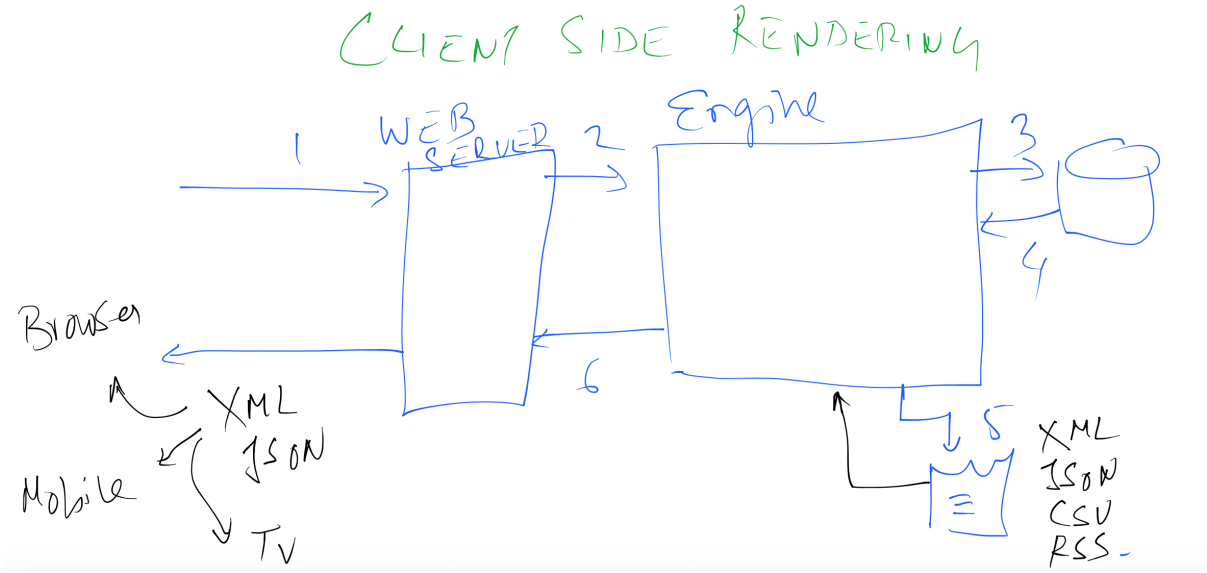
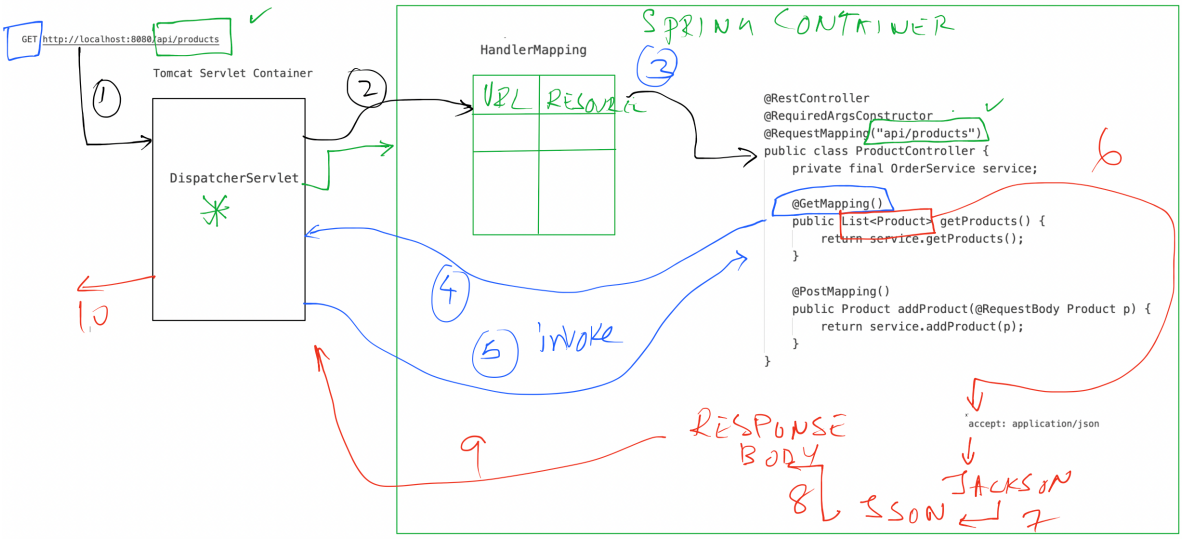
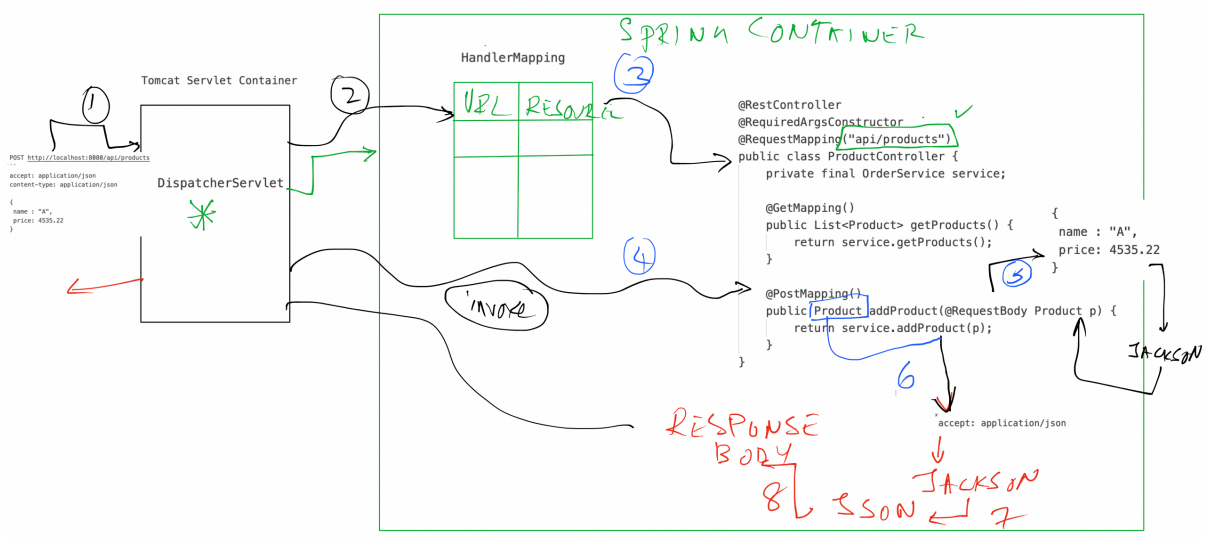

Spring Framework Runtime
Data Access/Integration
JDBC
ORM
OXM
JMS
Transactions
Web
WebSocket
Servlet
Web
Portlet
AOP
Aspects
Instrumentation
Messaging
Core Container
Beans
Core
Context
SpEL
Test
@Transactional


SERVER SIDE RENDERING
WEB SERVER
Engine
1
2
3
4
5
6
HTML
PDF
IMAGE
HTML
PDF
IMAGE


CLIENT SIDE RENDERING
WEB SERVER
Engine
Browser
Mobile
XML
JSON
TV
XML
JSON
CSV
RSS


SPRING CONTAINER
GET http://localhost:8080/api/products
Tomcat Servlet Container
DispatcherServlet
HandlerMapping
URL
RESOURCE
@RestController
@RequiredArgsConstructor
@RequestMapping("api/products")
public class ProductController {
private final OrderService service;
@GetMapping()
public List<Product> getProducts() {
return service.getProducts();
}
@PostMapping()
public Product addProduct(@RequestBody Product p) {
return service.addProduct(p);
}
}
accept: application/json
JACKSON
JSON
RESPONSE BODY
invoke

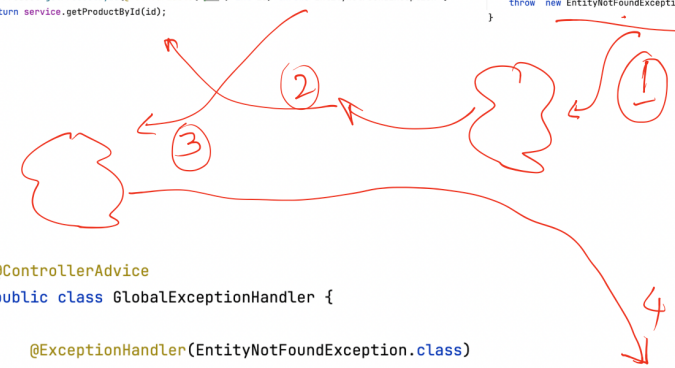

SPRING CONTAINER
Tomcat Servlet Container
DispatcherServlet
HandlerMapping
URL
RESOURCE
POST http://localhost:8080/api/products
accept: application/json
content-type: application/json
{
name : "A",
price: 4535.22
}
@RestController
@RequiredArgsConstructor
@RequestMapping("api/products")
public class ProductController {
private final OrderService service;
@GetMapping()
public List<Product> getProducts() {
return service.getProducts();
}
@PostMapping()
public Product addProduct(@RequestBody Product p) {
return service.addProduct(p);
}
}
invoke
JACKSON
accept: application/json
JACKSON
JSON
RESPONSE BODY

ProductController

```java
// http://localhost:8080/api/products/3
@GetMapping("/{pid}")
public Product getProductById(@PathVariable("pid") int id) throws EntityNotFoundException {
    return service.getProductById(id);
}
```

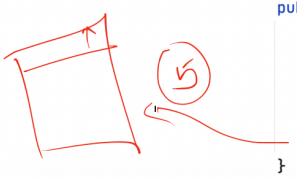
OrderService

```java
public Product getProductById(int id) throws EntityNotFoundException {
    Optional<Product> opt =  productDao.findById(id);
    if(opt.isPresent()) {
        return opt.get();
    }
    throw  new EntityNotFoundException("Product with id : " + id + " doesn't exist!!!");
}
```




```java
@ControllerAdvice
public class GlobalExceptionHandler {

    @ExceptionHandler(EntityNotFoundException.class)
    public ResponseEntity<Object> handleEntityNotFoundException(EntityNotFoundException ex) {
        Map<String, Object> data = new LinkedHashMap<>();
        data.put("timestamp", new Date());
        data.put("status", "404");
        data.put("message", ex.getMessage());
        return new  ResponseEntity<Object>(data, HttpStatus.NOT_FOUND); //404
    }
```




```
### @name="Add invalid Product"
POST http://localhost:8080/api/products
Accept: application/json
Content-Type: application/json

{
  "name": "",
  "price": 2.00,
  "quantity": 0
}
```

```java
public class Product {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @NotBlank(message="Name is required!!!")
    private String name;


    @Min(value = 10, message = "Price $... has to more than {value}")
    private  double price;

    @Min(value = 1, message = "Quantity $... has to more than {value}")
    @Column(name="qty")
    private int quantity;
```

MethodArgumentNotValidException

```java
@ExceptionHandler(MethodArgumentNotValidException.class)
public ResponseEntity<Object> handleMethodArgumentNotValidException(MethodArgum
    Map<String, Object> data = new LinkedHashMap<>();
    data.put("timestamp", new Date());
    List<String> errors = ex.getBindingResult() BindingResult
            .getFieldErrors().stream().map(exception -> exception.getDefaultMes
            .collect(Collectors.toList());
    data.put("errors", errors);
    return new  ResponseEntity<Object>(data, HttpStatus.BAD_REQUEST); //400
}
```

```java
@PostMapping()
@ResponseStatus(HttpStatus.CREATED) // 201
public Product addProduct(@RequestBody @Valid Product product) {
    return  service.addProduct(product);
}
```