* **@Prefix("/api")**

$data = $this->get('jms\_serializer')->serialize($article, 'json');

$response = new Response($data);

$response->headers->set('Content-Type', 'application/json');

return $response;

* **@Head("/booking)**
* **steps**
  + FindModel(*$booking*)

→ **Find the booking JSON Model**→ *return :* $bookingJsonModel =

* **Request-Data needed :** *Ressource named in the URI*
* **Response-Data sent:** *JSON Booking Model*
* **@Post("/booking/new/validate")**
* **Four steps**
  + Recept(*$request*)

→ **JSON Format *to* PHP Object** → *return :* srequest = PHP Object

* + Validate(*$request*)

→ **Entity Validation (PHP Object)** →*$request→ booking→ validate boolean =* true if Validate false if not valid ; *$request→ booking→ setMessages()* ; *$request→ booking→ setErrors()*→ *return :* $response

* + CreateResponse(*$response*)

*→*  **PHP Object to JSONFiles Encrypted Data →**return *$response* = JSON Object

* + return *$response*
* **Request-Data needed :** *Booking-data and Tickets-data in JSON format*
* **Response-Data sent:** *Booking-data and Tickets-data in JSON format*
  + With ⇒ new-data[ paiementStatus=> inProgress, validate => boolean true of false, messages => array[ ‘messages1’, ‘messages2’ ], errors =>array[ ‘errors1’, ‘errors2’] ]
* **@Post("/booking/new/save")**
* **Five steps**
  + Recept(*$request*)

→ **JSON Format *to* PHP Object** → *return :* $request = PHP Object

* + Save(*$request*)

→ **Entity Record (PHP Object)** →*$request→ booking→ saved→ boolean =* true if Validate false if not valid ; *$request→ booking→ setMessages()* ; *$request→ booking→ setErrors()*→ *return :* $response

* + mail($response, tosomeone@here.poc, template.twig)

→ **Send a mail to confirm payment and booking**

* + CreateResponse(*$response*)

*→*  **PHP Object to JSONFiles Encrypted Data →**return *$response* = JSON Object

* + return *$response*
* **Request-Data needed :** *Booking-data and Tickets-data in JSON format*
* With ⇒ new-data[ paiementStatus=> ok ]
* **Response-Data sent:** *Booking-data and Tickets-data in JSON format*
  + With ⇒ new-data[ paiementStatus=> ok, id=> booking-id, save => boolean true of false, messages => array[ ‘messages1’, ‘messages2’ ], errors =>array[ ‘errors1’, ‘errors2’] ]
* **@Get("/booking/show/{code}")**
* **Two steps**
  + Show(*$request*)

→ **Find a booking by code**→ *return :* $booking = PHP Object

* + render(*$booking*)

→ **Template rendering***→ booking→ {code}*

* **Request-Data needed :** *Booking-code*
* **Response-Data sent:** *Booking-data and Tickets-data in JSON format null if wrong code*
* **@Get("/booking/check-availability/ {date}")**
* **Three steps**
  + checkIfClosedDay(*$date*)

→ **Check if not a closed day**

* + CheckForAvailabiliy(*$date*)

→ **Check disponibility for the chosen day**→ *return :* $config = PHP Object

* + checkIfClosedDay(*$date*)

→ **Create a configuration files for the API***→ $config*

* **Request-Data needed :** *Request date*
* **Response-Data sent:** *Booking-API Configuration ⇒ $config Data in JSON format*
* **@Delete("/admin/booking/{code}")**
* **Two steps**
  + Find(*$request*)

→ **Find a booking by code**→ *return :* $booking = PHP Object

* + delete(*$booking*)

→ **Entity Delete (PHP Object)** *→ booking→ {code}*

* **Request-Data needed :** *Booking-code*
* **Response-Data sent:** *Messages for Booking-data and Tickets-data deletion info and Errors in JSON format*
* **@Options("/tickets")**
* **steps**
  + method(*$request*)

→ **XXXXXX**→ *return :* $ =

* **Request-Data needed :**
* **Response-Data sent:**