Scénario 1 : Vérifier la disponibilité d'une date

Step 1 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Choix de la date via datepicker
→ Choix du type, Journée ou Demi-Journée, via Checkbox
Si date du jour sélectionné et après 14 heures, seul la checkbox demi journée apparait, elle est déjà coché et en mode readOnly (impossible à modifier)

Step 2 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Conversion de la requête en JSON
→ Envois de la requête au serveur ⇒ Route : /booking/check-availability/{date}

Step 3 : Côté API Application (server) ⇒ Accès :/booking/check-availability/{date}
→ Reçoit la requête
→ *$config = Entity of Config (paramètre de l'application)*
→ Service : surikat_booking.availability
→ checkIfClosedDay($*closedDays*, $date) :
- *Vérifie si la date sélectionnée n'est pas dans la liste des dates fermées*
- *$closedDays = $config→ getClosedDays()* ⇒ *array*['date_value', 'date_value']
- $closedWeekDays = *$config→ getClosedWeekDay()* ⇒ *array*['day_value', 'day_value'] (close selected days of the week)

- *Si True ⇒ Config→ setErrors('Code erreur: 0001 - Information: Date non disponible à la réservation. - Cause : Jour de fermeture')*

RESPONSE 1 ⇒ *$*config → in JSON Format

- *Si False ⇒ chekForAvailability($date)*

→ checkAvailability($*ticketsCount* $*dailyTicketsLimit*)

- *Vérification de la disponibilité pour la date sélectionnée*
  → *$ticketsCount = le nombre de tickets déjà existant pour cette date = Sélection dans la BDD de toutes les réservations pour cette date, compte le total de tickets de ces réservations*
  → *$dailyTicketsLimit = $config→ getDailyTicketsLimit()*
  → *$availability = $dailyTicketsLimit -$dailyTicketsLimit*
  → *$config→ setAvailability($availability)*

- *Si $availability > 50 ⇒ $config→ setTicketsLimit() = 5*
  ⇒ $config→ setMessages() = 'Code LI02 Information: Limitation du nombre de tickets réservables à 5 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 50 tickets encore disponible'
- *Si $availability > 20 ⇒ $config→ setTicketsLimit() = 2*
  ⇒ $config→ setMessages() = 'Code LI03 Information: Limitation du nombre de tickets réservables à 2 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 20 tickets encore disponible'
- *Si $availability > 10 ⇒ $config→ setTicketsLimit() = 1*

⇒ $config→ setMessages() = 'Code LI04 Information: Limitation du nombre de tickets réservables à 1 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 5 tickets encore disponible'

- *Si $availability >= 0 ⇒ $config→ setTicketsLimit() = 0*
  ⇒ $config→ setErrors() = 'Code erreur: error_LI02 - Information: Date non disponible à la réservation. - Cause : Aucune disponibilité pour ce jour'

return: $config

RESPONSE 2 *⇒ $*config → in JSON Format

Step 4 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Réception de la requête en JSON
→ Configuration de l'application pour l'achat de ticket

→ → Direction Scenario 2

Scénario 2 : Acheter ses Tickets (Cas Disponibilité: OK)

Step 1 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Affichage de la date sélectionnée et du type de reservation
→ Affichage des messages d'info
→ Un premier ticket est déjà créé
→ Un bouton pour ajouter un ticket (si disponibilité supérieur à 1 et dans la limite définie)
→ Le formulaire à remplir pour chaque ticket
→ Le prix est mis à jour sur chaque billet (JavaScript ⇒ calcul depuis fichier de config reçu
→ Le Prix Total est Affiché

Step 2 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Conversion de la requête en JSON
→ Envois de la requête au serveur ⇒ Route : /booking/new

Step 3 : Côté API Application (server) ⇒ Accès :/booking/new

Reception :
→ Reçoit la requête
$booking = request-->getContent()

Validation :
→ Demande la validation de la requête via BookingController→ validate($booking)

Validation de la disponibilité :
→ chekForAvailability(*$booking-->getBookeddate)*) :

- *Vérifie si la date sélectionnée n'est pas dans la liste des dates fermées*
- *$closedDay* = *$config→ getClosedDay()* ⇒ *array*['date_value', 'date_value']
- $closedWeekDay = *$config→ getClosedWeekDay()* ⇒ *array*['day_value', 'day_value'] (close selected days of the week)

- *Si True ⇒ Booking→ setErrors('Code erreur: 0001 - Information: Date non disponible à la réservation. - Cause : Jour de fermeture')*
- *Si False ⇒ chekForAvailability($booking-->getBookeddate)*

$config = → chekForAvailability(*$booking-->getBookeddate*)

$config = Entity of Config (paramètre de l'application)

- *Vérifie si la disponibilité pour la date sélectionnée*
  → Sélectionne dans la BDD toutes les réservations pour cette date, compte le total de tickets de ces réservations
  → return $ticketsCount = le nombre de tickets déjà existant pour cette date
  → $dailyTicketsLimit = $config→ getDailyLimit()
  → $availability = $dailyTicketsLimit -$ticketCount
  → $config→ setAvailability($availability)

- *Si $availability > 50 ⇒ $config→ setTicketsLimit() = 5*
  ⇒ $config→ setMessages() = 'Code LI02 Information: Limitation du nombre de tickets réservables à 5 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 50 tickets encore disponible'
- *Si $availability > 20 ⇒ $config→ setTicketsLimit() = 2*
  ⇒ $config→ setMessages() = 'Code LI03 Information: Limitation du nombre de tickets réservables à 2 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 20 tickets encore disponible'
- *Si $availability > 10 ⇒ $config→ setTicketsLimit() = 1*
  ⇒ $config→ setMessages() = 'Code LI04 Information: Limitation du nombre de tickets réservables à 1 par réservation. - Cause : Disponibilité réduite pour ce jour : moins de 5 tickets encore disponible'
- *Si $availability >= 0 ⇒ $config→ setTicketsLimit() = 0*
  ⇒ $config→ setErrors() = 'Code erreur: error_LI02 - Information: Date non disponible à la réservation. - Cause : Aucune disponibilité pour ce jour'
- *return:* $config

$nbrOfTickets = → count($booking→ getTickets())

→ Calcul si le nombre de tickets demandés correspond à la disponibilité →
$disponibilityControl = ($config→ getAvailability()) - $nbrOfTickets

- *Si $disponibilityControl > 0 ⇒ priceValidator($booking) → priceValidator($booking)*
- *Si $disponibilityControl <= 0 ⇒ $booking→ setErrors('Code erreur: error_LI03 - Information: Tickets non disponible à la réservation. - Cause : Disponibilité inférieur à votre demande pour ce jour"*
- *- $booking→ setValidate(false)*

RESPONSE 1 *⇒ $booking in JSON format →* With ⇒ new-data[paiementStatus=> noStartedYet, validate => boolean false, messages => array['messages1', 'messages2'], errors =>array['errors1', 'errors2']]

Validation des tarifs et du prix total :

→ priceValidator(*$booking*)

- *$totalPrice= 0*
- *Pour chaque ticket :*
- *→ Récupère la date de naissance, le tarif enregistré et la valeur du boolean Special_Tarif (pour savoir si on applique le tarif réduit)*
- *→ Si Special_Tarif = true → $ticketPrice = $config→ getSpecialTarif*
- *→ Si Special_tarif = false →*

```
$datetime1 = new \DateTime(); // date actuelle

$datetime2 = new \DateTime('ticket.birthdate');

$age = $datetime1->diff($datetime2, true)->y;

// le y = nombre d'années ex : 22
```

- *$ticketPrice = calculPrice($age) (SERVICE) → return Integer value of the price calculate using Config entity Data*

- *$totalPrice = $totalPrice + $ticketPrice*

- *→ Vérifie si $ticketPrice et $ticket→ tarif correspondent*
    - *Si False*
        - *- $ticket→ setErrors('Code erreur: error_PRI02 - Information: Tarif non correspondant. - Cause : Le tarif enregistré pour votre ticket et le tarif réel fixé ne correspondent pas"*
        - *- $booking→ setValidate(false)*

    - *Si True*
        - *→ Vérifie si $totalPrice et $booking→ totalPrice correspondent*
            - *Si False*
                - *- $booking→ setErrors('Code erreur: error_PRI01 - Information: Tarif non correspondant. - Cause : Le tarif enregistré pour votre réservation et le tarif réel caclulé ne correspondent pas"*
                - *- $booking→ setValidate(false)*
            - *Si True*
                - *- $booking→ setValidate(true)*

→ return($booking)

- *SI $booking→ getValidate = false*

RESPONSE 2 *⇒ $booking in JSON format →* With ⇒ new-data[paiementStatus=> noStartedYet, validate => boolean false, messages => array['messages1', 'messages2'], errors =>array['errors1', 'errors2']

- *SI $booking→ getValidate = true*

  → getPaiement($booking)

  SI $booking→ getPaiementStatus = ok
  $booking→ setMessages('Réservation enregistré et paiement réalisé avec succès)
  → save($booking)

RESPONSE 3 *⇒ $booking in JSON format →* With ⇒ new-data[paiementStatus=> OK, validate => boolean true, messages => array['messages1', 'messages2'], errors =>array['errors1', 'errors2']]]

- *SI $booking→ getPaiementStatus = failed*
  $booking→ setErrors(‘Code erreur: error_PAY01 - Information: Paiement non pris en compte - Cause : Le processus de paiement à rencontré une erreur, veuillez recommencé”)

RESPONSE 4 ⇒ *$booking in JSON format* → With ⇒ new-data[paiementStatus=> failed, validate => boolean true, messages => array[‘messages1’, ‘messages2’], errors =>array[‘errors1’, ‘errors2’]]]

Dans tos les cas, envois d’un mail avec les informations sur la réservation

Step 4 : Côté Frontend Application (client) ⇒ Accès : /booking
→ Réception des données
→ Si erreur : Affichage des erreurs en vue d’apporter les modifications nécessaires et nouvelle soumission
→ Si tout es ok : Affiche la réservation et informe de sa validation

Scénario 3 : Rechercher sa Réservation

Step 1 : Côté Frontend Application (client) ⇒ Accès : /booking/show
→ Affichage du moteur de recherche par code
→ envois de la requête ⇒ Route = /booking/show/{code}

Step 2 : Côté API Application (server) ⇒ Accès :/booking/check-availability/{date}
→ Réception des données
→ Interroge la base de donnée avec findByCode(*$code*)
→ Si $date n’existe pas

RESPONSE 1 ⇒ *$booking in JSON format → with Null Content*

→ Si $date existe

RESPONSE 2 ⇒ *$booking in JSON format* → with content (sans les infos de la carte pour le paiement qui elle ne sont jamais enregistrées)

Step 3 : Côté Frontend Application (client) ⇒ Accès : /booking/show
→ Affichage de la Réservation