package Droneshotsbooking.controller;

import java.util.Date;

import java.util.List;

import java.util.Optional;

import javax.annotation.PostConstruct;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.cloud.client.loadbalancer.Response;

import org.springframework.http.HttpHeaders;

import org.springframework.http.MediaType;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import org.springframework.web.client.RestTemplate;

import org.springframework.web.reactive.function.client.WebClient;

import Droneshotsbooking.pojo.Booking;

import Droneshotsbooking.pojo.Drone;

import Droneshotsbooking.pojo.DroneData;

import Droneshotsbooking.pojo.customer;

import Droneshotsbooking.repository.BookingRepository;

import Droneshotsbooking.service.EmailService;

import lombok.extern.slf4j.Slf4j;

import reactor.core.publisher.Mono;

@RestController

@RequestMapping("/Droneshotsbooking")

@Slf4j

@CrossOrigin(origins = "\*", maxAge = 3600)

public class bookingController {

@Autowired

public BookingRepository book\_repo;

@Autowired

public WebClient.Builder webClientBuilder;

@Autowired

private RestTemplate restTemplate;

@Autowired

EmailService emailService;

/\*

\* @GetMapping("/find") public List<DroneData> findDroneData() { Drone drone

\* = restTemplate.getForObject("http://allDrones", Drone.class); return

\* drone.getdrone(); }

\*/

/\*

\* @RequestMapping("/find") public Response getResponse() {

\* ResponseEntity<Response> responseEntity =

\* restTemplate.getForEntity("http://allDrones", Response.class);

\*

\* return responseEntity.getBody(); }

\*/

// For user to book Drone and amount being set according to passenger shots details

@PostMapping("/BookDrone/{mailId}")

public String bookDrone(@RequestBody Booking booking,@PathVariable("mailId") String mailId)

{ log.info("Making the booking with pnr:"+booking.getBooking\_id());

booking.setTotal\_amount(setTotal\_amount(booking));

boolean present=book\_repo.findById(booking.booking\_id).isPresent();

if(present==false) {

log.info("Made the booking with pnr:"+booking.getBooking\_id());

book\_repo.save(booking);

/\* return "Booking made with PNR:"+booking.getBooking\_id(); \*/

return emailService.sendEmail(booking,mailId);}

log.error("Booking Already present");

return"Booking Already Made";

}

// For User+Attendee to get Details of booking (booking\_id=PNR)

@GetMapping("/BookedDrone/{booking\_id}")

public Optional<Booking> getBooking (@PathVariable("booking\_id") long booking\_id){

try { log.info("getting Ticket: "+booking\_id);

return book\_repo.findById(booking\_id);}

catch(Exception e){

log.error(e.toString());

return book\_repo.findById(booking\_id) ;}

}

//error:-string to date

//Admin to get all bookings on the date

@GetMapping("/Booking/{date}/{drone\_id}")

public List<Booking>getBookingByDate(@PathVariable Date date,@PathVariable int drone\_id){

log.info("getting Ticket: "+drone\_id);

return book\_repo.findByDate(date);

}

// For user to Cancel Booking OR Self Check-In

@PutMapping("/booking/{booking\_id}/{mailId}")

public Booking updateBooking(@RequestBody Booking booking,@PathVariable("booking\_id") long pnr,@PathVariable("mailId") String mailId) {

Booking dbResponse=book\_repo.findById(pnr).get();

dbResponse.setBooking\_cancelled(booking.isBooking\_cancelled());

dbResponse.setChecked\_in(booking.isChecked\_in());

log.info("Saving Changes In Booked shots Of Booking Cancelled/ CheckIn: "+pnr);

book\_repo.save(dbResponse);

if(booking.checked\_in==true) {

System.out.println("checked in");

log.info("Updated CheckIn Status");

return emailService.checkIn(dbResponse,mailId);}

if(booking.booking\_cancelled==true)

{booking.setChecked\_in(false);

System.out.println("Booking Cancelled");

log.info("Updated Booking Cancelled Status");

return emailService.bookingCancelled(dbResponse,mailId);}

return dbResponse;

}

//Setting total amount for payment gateway fetched from customer data

public double setTotal\_amount(Booking booking) {

double tot\_sum=0;

List<Customer> list= booking.getPassenger();

for (Customer customer : list) {

tot\_sum+=customer.getAmount();

}

return tot\_sum;

}

}