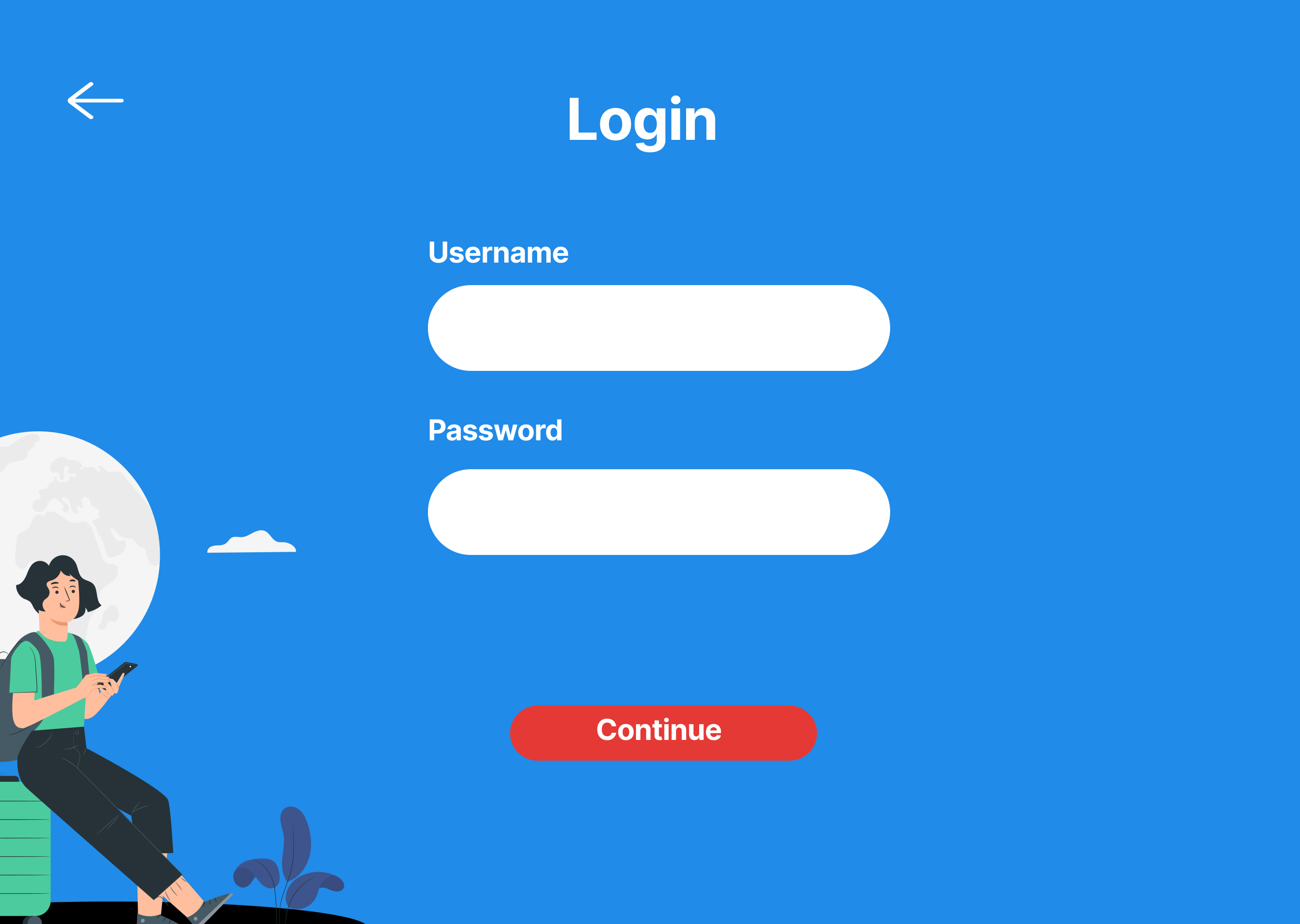
DOCUMENTATION- BUS RESERVATION SYSTEM

Login For Customer:



@PostMapping("/add")

**public** Customer insertCustomer(@RequestBody Customer customer) {

//save user info in db

User user=customer.getUser();

// i am encrypting the password

String passwordPlain = user.getPassword();

String encodedPassword = passwordEncoder.encode(passwordPlain);

user.setPassword(encodedPassword);

user.setRole("CUSTOMER");

user = userService.insert(user);

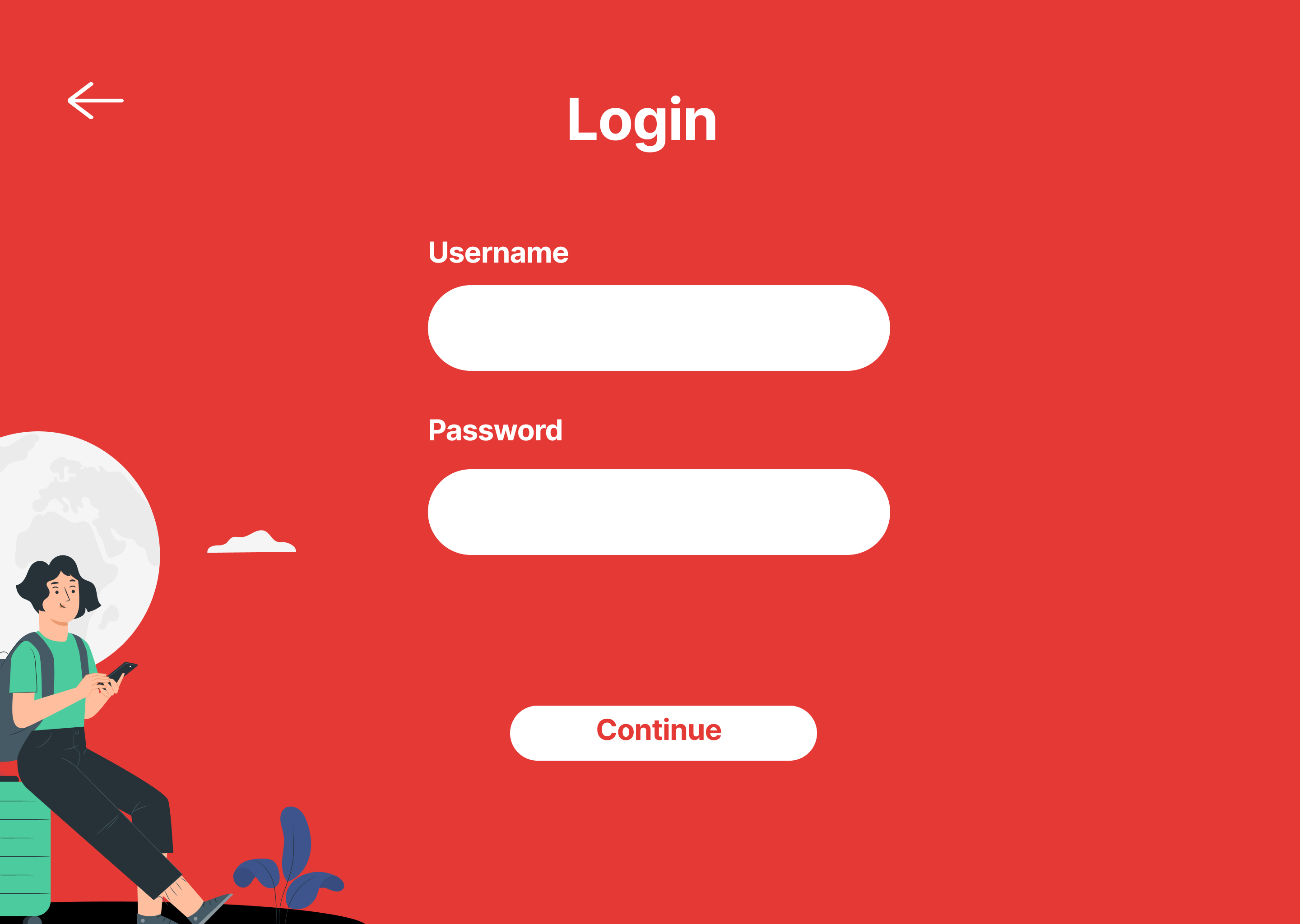
// attach the saved user(in step 1)

customer.setUser(user);

**return** customerService.insert(customer);

}

Login For BusOperator:



@PostMapping("/add/{eid}")

**public** ResponseEntity<?> assignBusOperator(@PathVariable("eid") **int** eid,@RequestBody BusOperator busOperator) {

**try** {

Executive executive=executiveService.getById(eid);

busOperator.setExecutive(executive);

//save user info in db

User user=busOperator.getUser();

// i am encrypting the password

String passwordPlain = user.getPassword();

String encodedPassword = passwordEncoder.encode(passwordPlain);

user.setPassword(encodedPassword);

user.setRole("BUSOPERATOR");

user = userService.insert(user);

// attach the saved user(in step 1)

busOperator.setUser(user);

busOperator=busOperatorService.insert(busOperator);

**return** ResponseEntity.*ok*().body(busOperator);

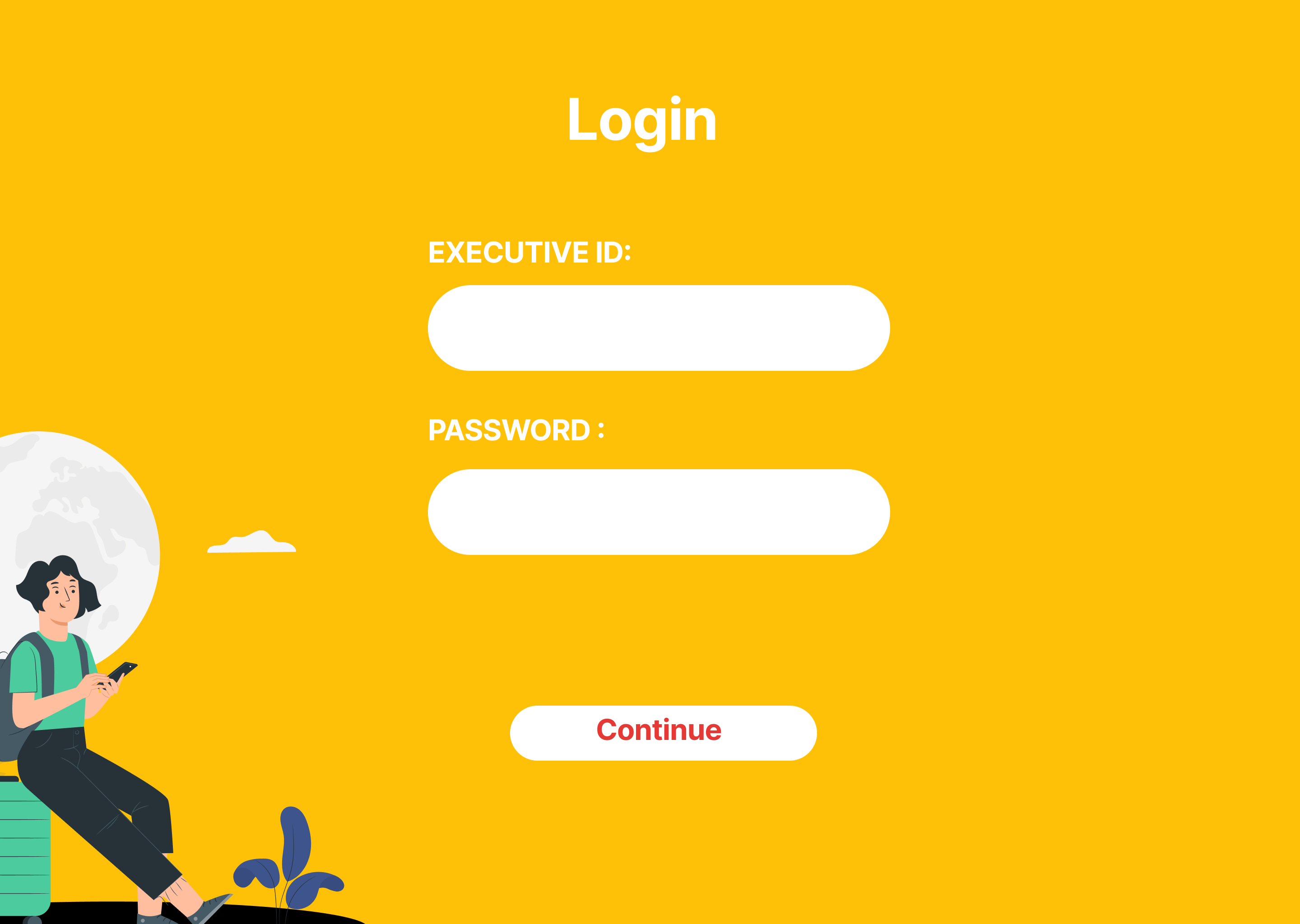
}**catch**(InvalidIdException e){

**return** ResponseEntity.*badRequest*().body(e.getMessage());

}

}

Login For Executive:



@PostMapping("/add")

**public** Executive insertExecutive(@RequestBody Executive executive) {

User user=executive.getUser();

String passwordPlain = user.getPassword();

String encodedPassword = passwordEncoder.encode(passwordPlain);

user.setPassword(encodedPassword);

user.setRole("EXECUTIVE");

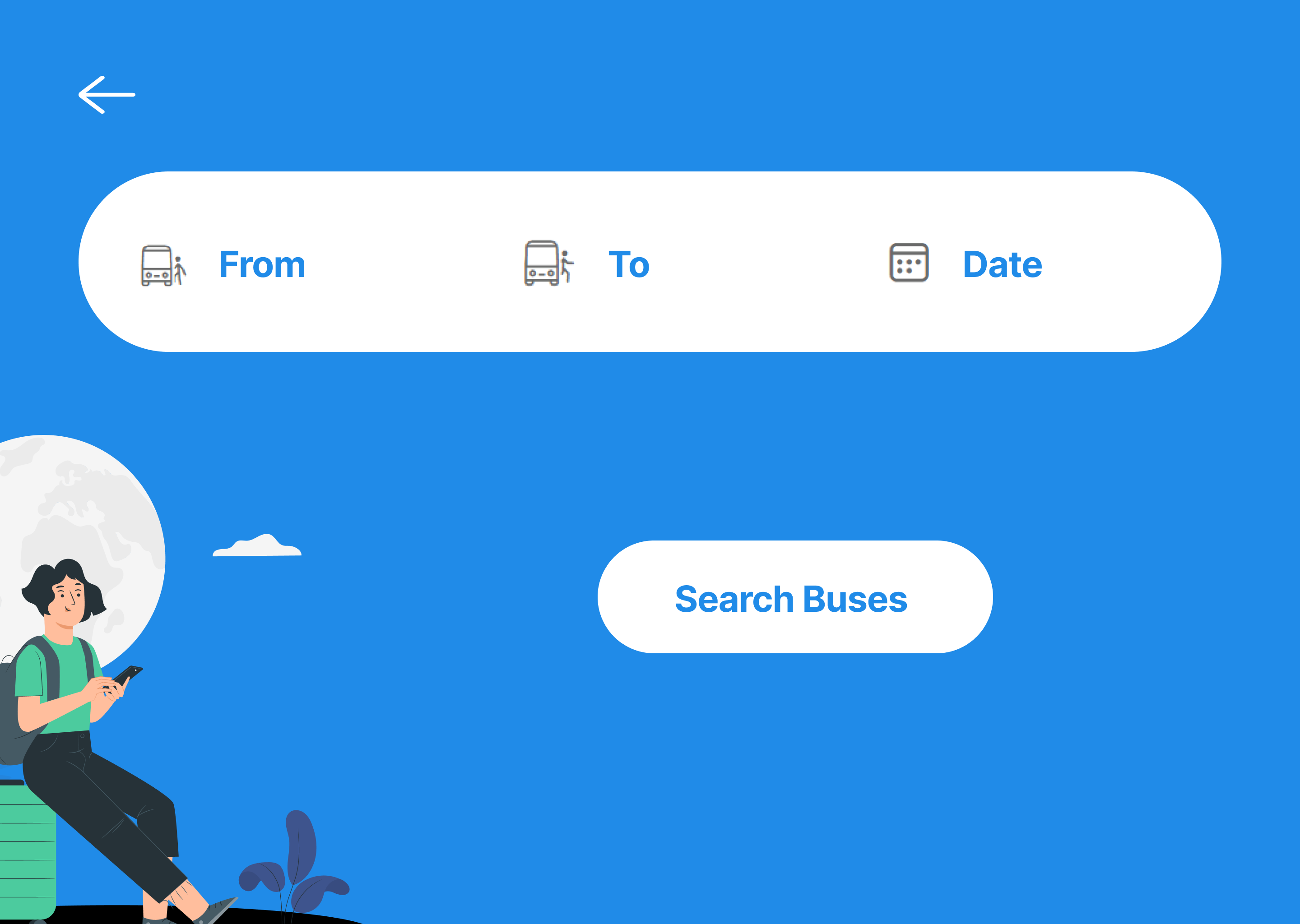
user = userService.insert(user);

executive.setUser(user);

**return** executiveService.insert(executive);

}

Search Bus:



@GetMapping("/get/{source}/{destination}/{date}")

**public** List<Bus> getBussesbysdd(@PathVariable("source") String source,

@PathVariable("destination") String destination,

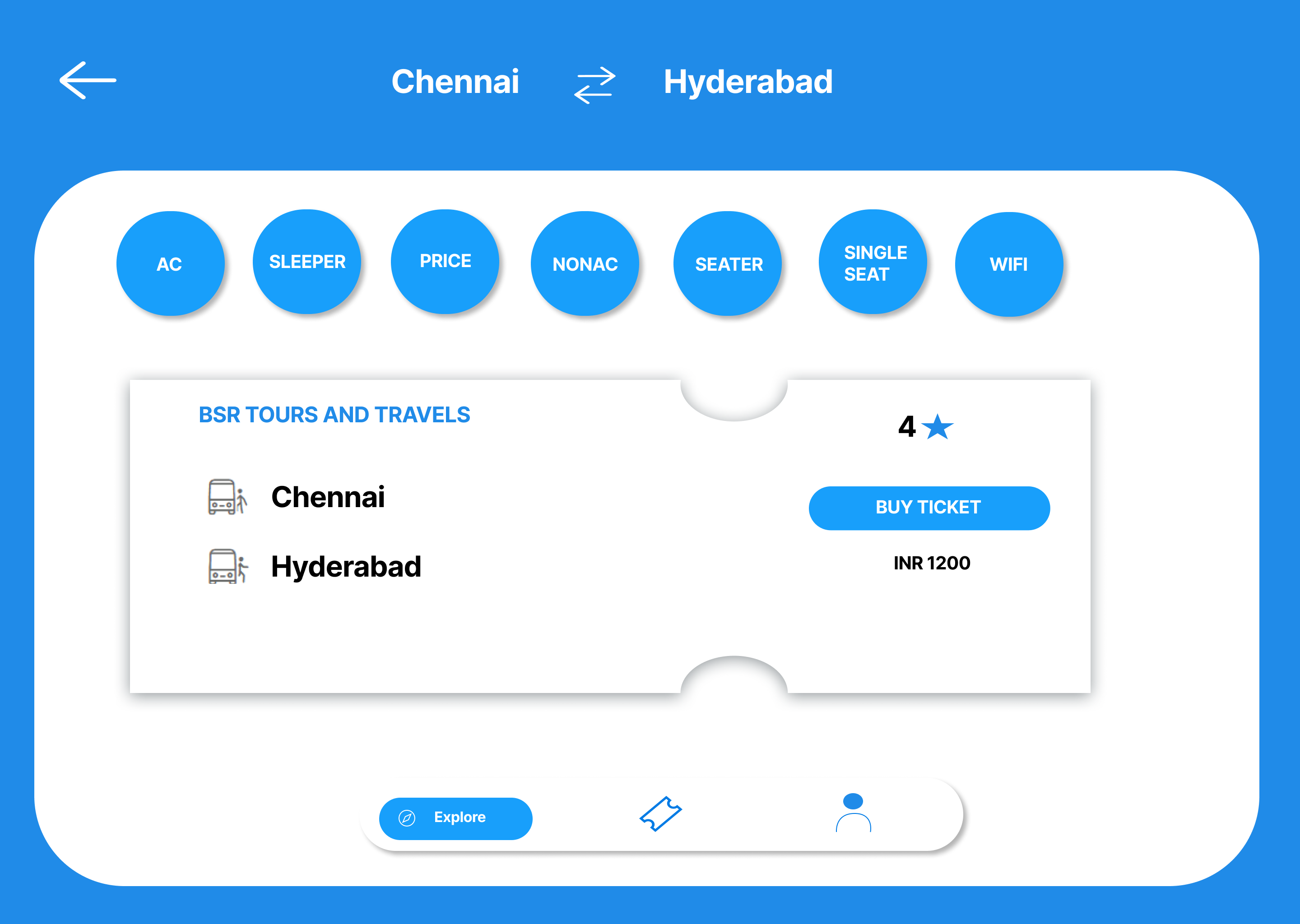
@PathVariable("date") @DateTimeFormat(iso = DateTimeFormat.ISO.***DATE***) LocalDate date) {

List<Bus> list = busService.findby(source, destination, date);

**return** list;

}

5. Select Bus:



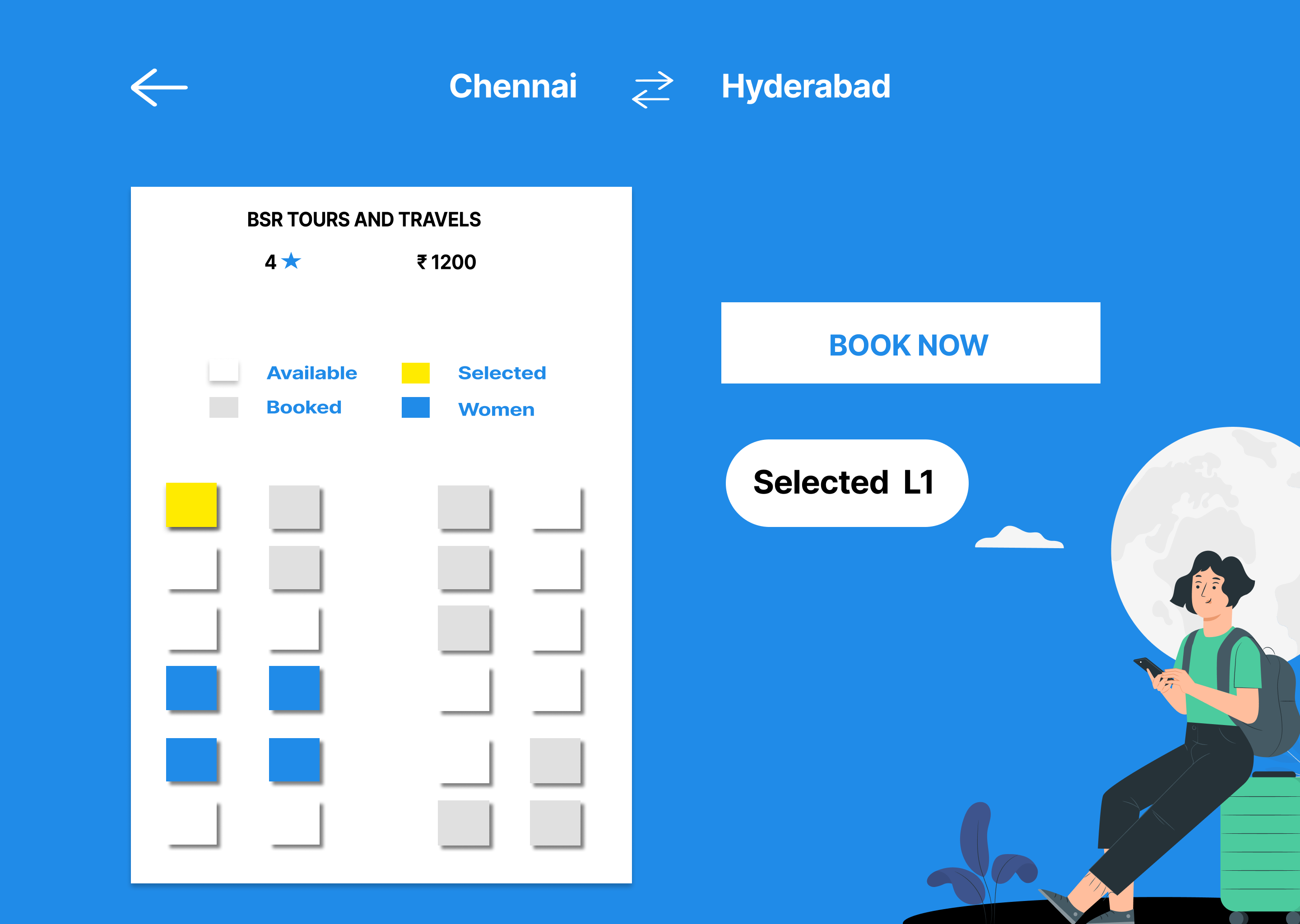
@GetMapping("/get/{bid}")

**public** List<Bus> getBusById(@PathVariable("bid") **int** bid){

**return** busService.getBusById(bid);

}

Select Seats:



@PostMapping("/booking/multiple/{cid}/{bid}")

**public** ResponseEntity<?> bookTickets(@PathVariable("cid") **int** cid, @PathVariable("bid") **int** bid,

@RequestBody List<PassengerDto> passengerDtoList) {

**try** {

Customer customer = customerService.getById(cid);

Bus bus = busService.getById(bid);

List<CustomerBus> bookedTickets = **new** ArrayList<>();

**double** totalPrice=0;

**for** (PassengerDto passengerDto : passengerDtoList) {

CustomerBus customerBus = **new** CustomerBus();

BusSchedule busSchedule =busScheduleService.getByBusId(bid);

customerBus.setCustomer(customer);

customerBus.setBus(bus);

customerBus.setDateOfBooking(LocalDate.*now*());

// Set passenger-specific information from the DTO

customerBus.setPassengerName(passengerDto.getPassengerName());

customerBus.setAge(passengerDto.getAge());

customerBus.setGender(passengerDto.getGender());

customerBus.setPrice(busSchedule.getFare());

seat.setSeatNumber(passengerDto.getSeatNo());

String seatNo=passengerDto.getSeatNo();

customerBusService.changestatus(seatNo);

totalPrice = totalPrice+(customerBus.getPrice());

// Add the processed ticket to the list

bookedTickets.add(customerBusService.insert(customerBus));

}

Map<String, Object> response = **new** HashMap<>();

response.put("bookedTickets", bookedTickets);

response.put("totalPrice", totalPrice);

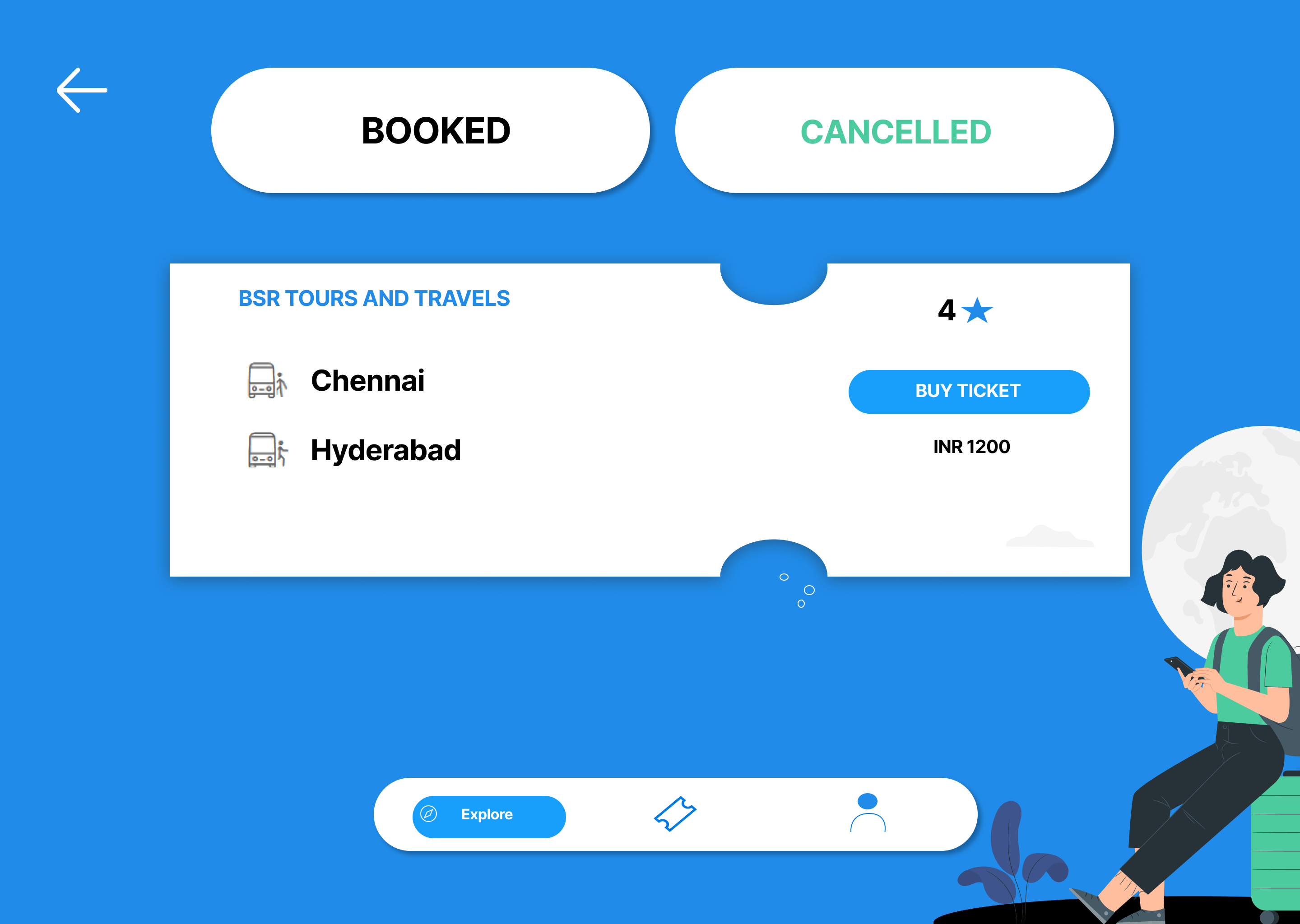
**return** ResponseEntity.*ok*().body(response);

} **catch** (InvalidIdException e) {

**return** ResponseEntity.*badRequest*().body(e.getMessage());

}

}Booked Status:



@GetMapping("/bookings/{cid}")

**public** List<Customer> getAllCustomer(

@RequestParam(value = "page", required = **false**, defaultValue = "0") Integer page,

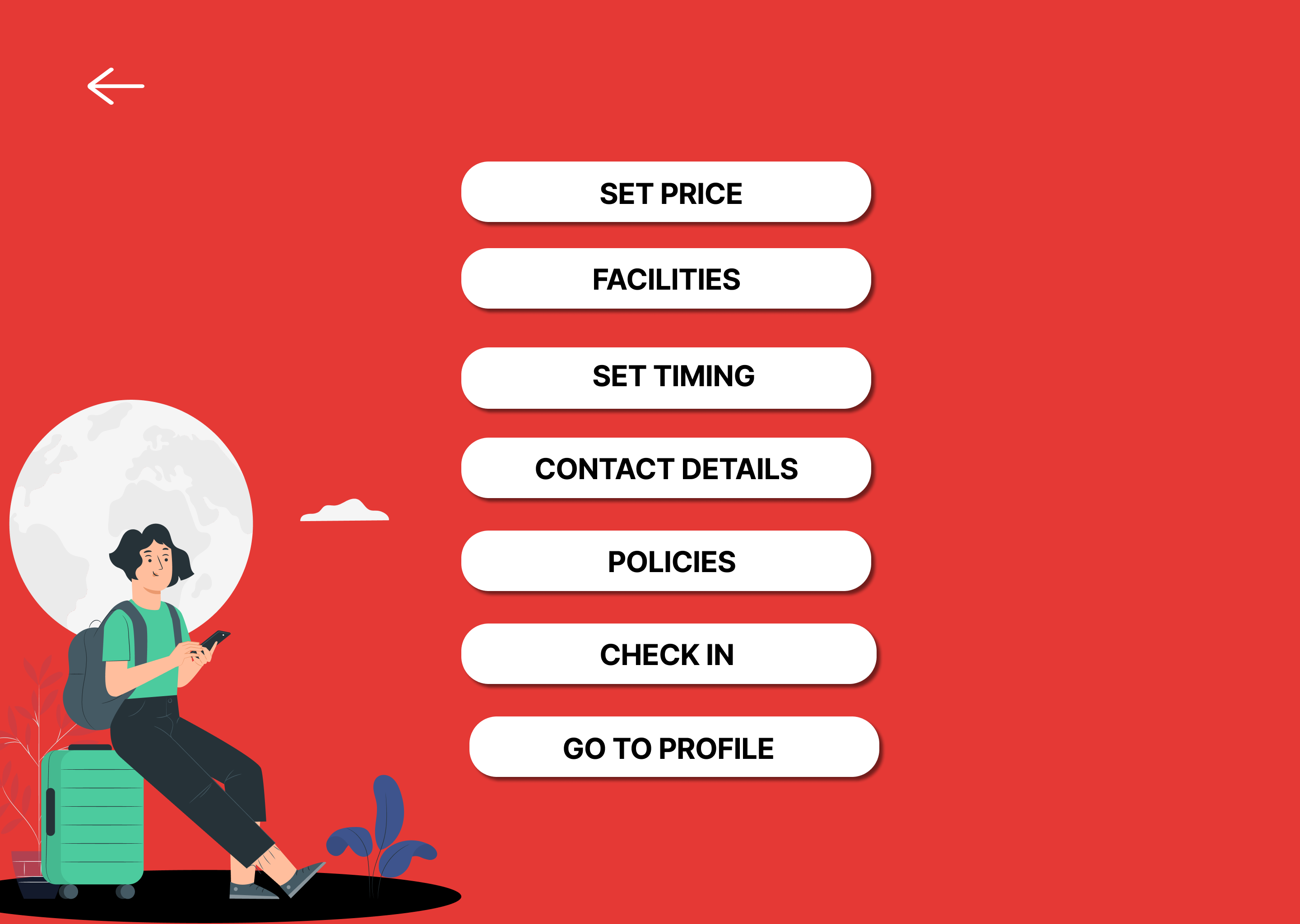
@RequestParam(value = "size", required = **false**, defaultValue = "10000000") Integer size) {

Pageable pageable = PageRequest.*of*(page, size); // null null

**return** customerService.getAll(pageable);

}

BusOperator Bus Add:



@PostMapping("/add/{bid}/{boid}")

**public** ResponseEntity<?> assignBus (@PathVariable("boid")**int** boid,@PathVariable("bid")**int** bid,

@RequestBody BusSchedule busSchedule){

**try** {

BusOperator busOperator=busOperatorService.getById(boid);

Bus bus=busService.getById(bid);

busSchedule.setBusOperator(busOperator);

busSchedule.setBus(bus);

busSchedule=busScheduleService.insert(busSchedule);

**return** ResponseEntity.*ok*().body(busSchedule);

}

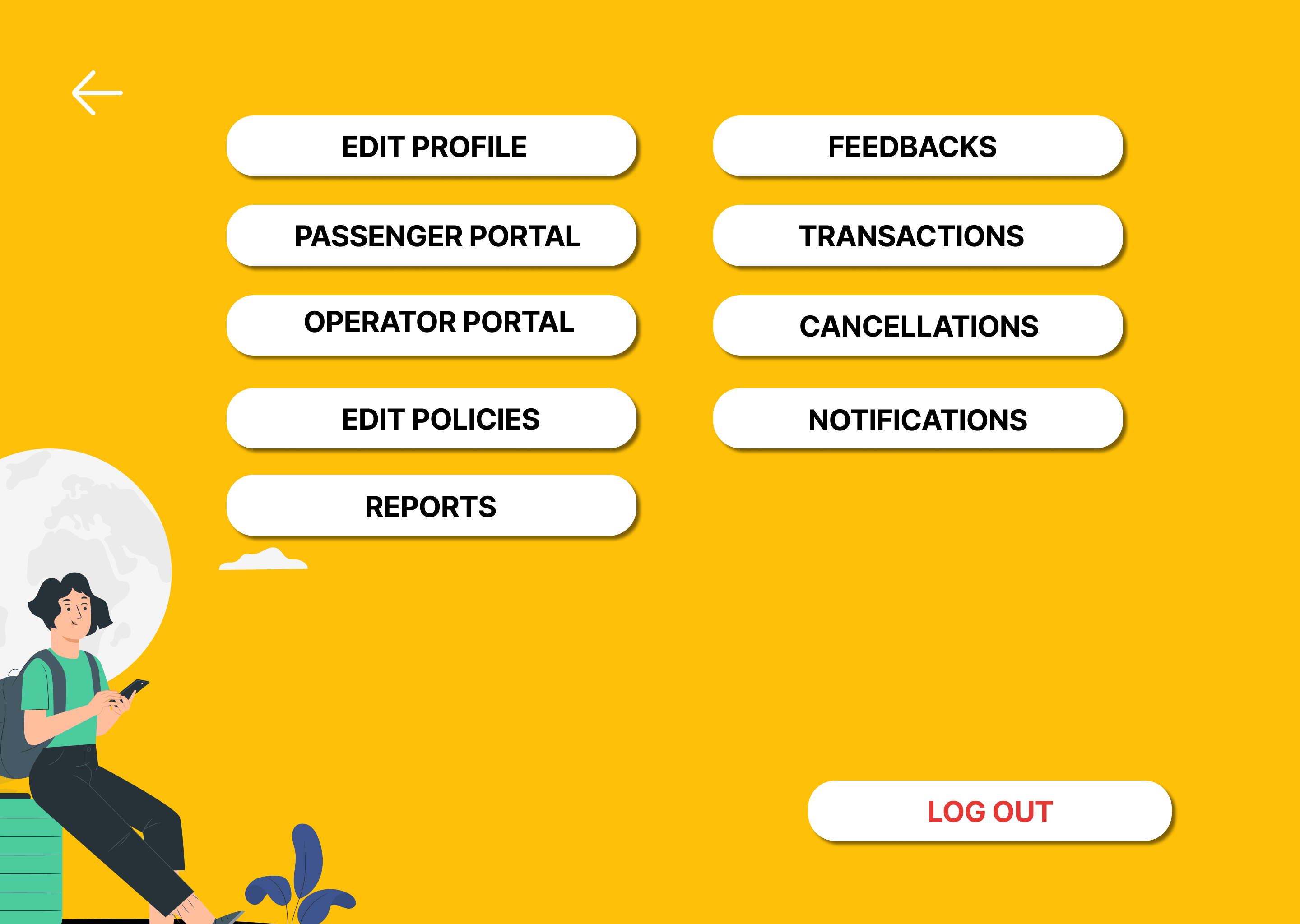
**catch**(InvalidIdException e) {

**return** ResponseEntity.*badRequest*().body(e.getMessage());

}

}

Executive Update/delete/Get:



@GetMapping("/get/bus/{eid}")

**public** List<Bus> getBusByExecutiveId(@PathVariable("eid") **int** eid){

**return** busService.getBusByExecutiveId(eid);

}

@GetMapping("/get/busOperator/{eid}")

**public** List<BusOperator> getBusOperatorByExecitiveId(@PathVariable("eid") **int** eid){

**return** busOperatorService.getBusOperatorByExecitiveId(eid);

}

@GetMapping("/get/customer")

**public** List<Customer> getCustomer(){

**return** customerService.getCustomer();

}

@PutMapping("/update/{eid}")

**public** ResponseEntity<?> updateExecutive(@PathVariable("eid") **int** eid,@RequestBody ExecutiveDto executiveDto){

**try** {

Executive executive = executiveService.getById(eid);

**if**(executiveDto.getPhoneNo()!=**null**)

executive.setPhoneNo(executiveDto.getPhoneNo());

**if**(executiveDto.getName()!=**null**)

executive.setName(executiveDto.getName());

**if**(executiveDto.getEmail()!=**null**)

executive.setEmail(executiveDto.getEmail());

executive=executiveService.postExecutive(executive);

**return** ResponseEntity.*ok*().body(executive);

}**catch**(InvalidIdException e) {

**return** ResponseEntity.*badRequest*().body(e.getMessage());

}

}



