Restful API

http://localhost:8080/AgileSpace-war/webresources/

Legend:
P - pass
JPQL - jpql needs fixing
NULL - nullpointerexception causing internal error 500.

Customers

Status	Type	End Points	Description
P	POST	/customers	Create a new Customer by using the request body data. Should return back the customer object (with id value).
P	GET	/customers/query?all=active	Get all customers whose account are active
P	GET	/customers	Get all customers account
P	GET	/customers/query?email=	Get customer whose email =
P	GET	/customers/query?firstName=	Get all customers with name containing “___” (case-insensitive)
P	GET	/customers/query?lastName=	Get all customers with name containing “___” (case-insensitive)
P	GET	/customers/query?phoneNum=	Get all customers with phone containing “___” (case-insensitive)
P	GET	/customers/query?rating=	Get all customers with phone containing “___” (case-insensitive)
P	POST	/customers/login	Login with email and hashed password
P	GET	/customers/10	Get Customer with id = 10
P	PUT	/customers/10	Edit Customer with id = 10 using the request body data (only basic attributes: email, password, firstName, lastName, phoneNum, entityStatus, rating). Should return empty payload if the update is success or 404 if the customer is not found.

P	DELETE	/customers/10	Delete Customer with id = 10
P	POST	/customers/10/url	Add a url (using the request body data) to Customer with id = 10. Should return back the customer object (with id value).
NotMarkedWIthCascadePersist	POST	/customers/10/listing	Add a Listing (using the request body data) to Customer with id = 10. Should return back the customer object (with id value). Bug: java.lang.IllegalStateException: During synchronization a new object was found through a relationship that was not marked cascade PERSIST: entity.Listing TODO: After listing.AssignCustomer(), call customer.getListings to check for update.
	POST	/customers/10/booking	Add a Booking (using the request body data) to Customer with id = 10. Should return back the customer object (with id value).
	POST	/customers/10/feedback	Add a feedback (using the request body data) to Customer with id = 10. Should return back the customer object (with id value).
	POST	/customers/10/hostDoorSystem	Add a hostDoorSystem (using the request body data) to Customer with id = 10. Should return back the customer object (with id value).
Bad Request	DELETE	/customers/10/url	Delete the url from Customer (with id = 10)
P	DELETE	/customers/10/listing/123	Delete the association between Customer (with id = 10) and Field (with id = 123). Then, delete the Listing record if there is no more association

			with any other Customer after this association deletion.
P	DELETE	/customers/10/booking/123	Delete the association between Customer (with id = 10) and Field (with id = 123). Then, delete the Listing record if there is no more association with any other Customer after this association deletion.
P	DELETE	/customers/10/feedback/123	Delete the association between Customer (with id = 10) and Field (with id = 123). Then, delete the Listing record if there is no more association with any other Customer after this association deletion.
P	DELETE	/customers/10/hostDoorSystem/123	Delete the association between Customer (with id = 10) and Field (with id = 123). Then, delete the Listing record if there is no more association with any other Customer after this association deletion.

Booking

Type	**End Points**	**Description**	**Normal**	**Validate**
POST	/bookings/customer/123/slot/123	Create a new Booking by using the request body data. Should return back the booking object (with id value). Do association of customer and slots to booking here	O	
GET	/bookings/10	Get Booking with id = 10	o	
GET	/bookings	Get all bookings	o	
GET	/bookings/query?qrcode=	Get Booking by QRCode	o	

PUT	/bookings/10	Edit Booking with id = 10 using the request body data (only basic attributes: qrCode, bookingStatusEnum, EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.	o	
DELETE	/bookings/10	Delete Booking with id = 10	O	
POST	/bookings/10/slot	Add a slot (using the request body data) to Booking with id = 10. Should return back the booking object (with id value).	O	
GET	/bookings/slot	Search Booking using Slot object's attributes using the request body data (only basic attributes: startDate, endDate, availability, entityStatusEnum).	-	
DELETE	/bookings/10/slot/123	Delete the association between Booking (with id = 10) and Slot (with id = 123). Then, delete the Slot record if there is no more association with any other Booking after this association deletion.	O	

Feedback

Type	End Points	Description	Normal	Validate

POST	/customer/10/listing/10	Create a new feedback by using the request body data. Should return back the customer object (with id value). Do association of customer and listing to feedback here	o	
GET	/feedbacks/10	Get feedback with id = 10	O	
GET	/feedbacks	Get all feedbacks	O	
GET	/feedbacks/query?feedbackType=	Get all feedbacks with feedbackType	O	
GET	/feedbacks/query?rating=	Get all feedbacks with rating	x	
GET	/feedbacks/query?remarks=	Get all feedbacks with remarks	x	
GET	/feedbacks/query?entityStatusEnum=	Get all feedbacks with entityStatusEnum	x	
PUT	/feedbacks/10	Edit feedback with id = 10 using the request body data (only basic attributes: feedbackType, rating, remarks,EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.	O	
DELETE	feedbacks/10	Delete feedback with id = 10	O	

POST	/feedbacks/10/customer	Assign a customer who gave the feedback (using the request body data) with id = 10. Should return back the feedback object (with id value).	O	
POST	/feedbacks/10/listing	Assign a listing to the feedback (using the request body data) with id = 10. Should return back the feedback object (with id value).	O	
DELETE	/feedbacks/10/customer/123	Delete the association between Feedback (with id = 10) and Customer (with id = 123). Then, delete the Listing record if there is no more association with any other Feedback after this association deletion.	O	
DELETE	/feedbacks/10/listings/123	Delete the association between Feedback (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other feedback after this association deletion.	O	

HostDoorSystem

Type	End Points	Description	Normal	Validate

POST	/customer/{customerid}/listing/{listingsId}	Create a new hostDoorSystem by using the request body data. Should return back the customer object (with id value). Do association of customer and listing to hostDoorSystem here	O .	Validation yet to be placed back
GET	/hostDoorSystems/10	Get hostDoorSystem with id = 10	o	
GET	/hostDoorSystems	Get all hostDoorSystems	o	
GET	/hostDoorSystems/query?email=	Get hostDoorSystem by email	O	
POST	/hostDoorSystems/login	Login with email and hashed password	O	Yet to be encrypted Send json object
PUT	/hostDoorSystems/10	Edit Booking with id = 10 using the request body data (only basic attributes: qrCode, bookingStatusEnum, EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.	Works but without validation due to search error	
DELETE	/hostDoorSystems/10	Delete hostDoorSystem with id = 10	o	
DELETE	/hostDoorSystems/10/customer/123	Delete the association between hostDoorSystem (with id = 10) and Customer (with id = 123). Then, delete the Customer record if there is no more association with any other hostDoorSystem after this association deletion.	o	Customer is not deleted
DELETE	/hostDoorSystems/10/listi	Delete the association between	o	Listing is

	ng/123	hostDoorSystem (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other hostDoorSystem after this association deletion.		will not deleted

Listings

S/N	Type	End Points	Description
P	POST	/listings/customer/123	Create a new Listing by using the request body data. Should return back the listing object (with id value).
P	GET	/listings	Get all listings
P	GET	/listings/10	Get Listing with id = 10
P	GET	/listings/query?buildingType=	Get all listings with buildingType
P	GET	/listings/query?header=	Get all listings with header
P	GET	/listings/query?description=	Get all listings with description
	GET	/listings/query?price=	Get all listings with price
	GET	/listings/query?unitLevel=	Get all listings with unitLevel
	GET	/listings/query?unitNumber=	Get all listings with unitNumber
P	GET	/listings/query?entityStatusEnum=	Get all listings with entityStatusEnum
P	PUT	/listings/10	Edit Listing with id = 10 using the request body data (only basic attributes: buildingType, header, description, price, unitLevel, unitNumber, entityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.
P	DELETE	/listings/10	Delete Listing with id = 10
P	POST	/listings/10/customer	Assign a customer (using the request body data) to listing with id = 10. Should

			return back the listing object (with id value).
	POST	/listings/10/feedback	Add a Feedback (using the request body data) to Listing with id = 10. Should return back the listing object (with id value).
P	POST	/listings/10/slot	Add a slot (using the request body data) to listing with id = 10. Should return back the customer object (with id value).
P	POST	/listings/10/hostDoorSystem	Add a hostDoorSystem (using the request body data) to listing with id = 10. Should return back the listing object (with id value).
P	POST	/listings/10/location	Add a Location(using the request body data) to Listing with id = 10. Should return back the listing object (with id value).
P	DELETE	/listings/10/customer/123	Delete the association between listing (with id = 10) and customer (with id = 123). Then, delete the Listing record if there is no more association with any other customer after this association deletion.
P	DELETE	/listings/10/feedback/123	Delete the association between Listing (with id = 10) and feedback (with id = 123). Then, delete the Listing record if there is no more association with any other feedback after this association deletion.
P	DELETE	/listings/10/slot/123	Delete the association between Listing (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other slot after this association deletion.
P	DELETE	/listings/10/hostDoorSystem/123	Delete the association between

			listing (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other hostDoorSystem after this association deletion.
P	DELETE	/listings/10/location/123	Delete the association between hostDoorSystem (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other Location after this association deletion.

Location

S/N	Type	End Points	Description
P	POST	/locations/listing/123	Create a new location by using the request body data. Should return back the location object (with id value). Do association of customer and listing to feedback here
P	GET	/locations/10	Get locations with id = 10
P	GET	/locations	Get all locations
	GET	/locations/query?address=	Get all locations with address
	GET	/locations/query?postalCode=	Get all locations with postalCode
	GET	/locations/query?longtitude=	Get all locations with longtitude
	GET	/locations/query?latitude=	Get all locations with latitude
	GET	/locations/query?entityStatusEnum=	Get all locations with entityStatusEnum
P	PUT	/locations/10	Edit location with id = 10 using the request body data (only basic attributes: address, postalCode, longtitude, latitude, EntityStatusEnum). Should return empty payload if the update is success or 404 if the

			customer is not found.
	DELETE	/locations/10	Delete location with id = 10
	POST	/locations/10/listing	Assign a listing who gave the feedback (using the request body data) with id = 10. Should return back the customer object (with id value).
P	DELETE	/locations/10/listing/123	Delete the association between location (with id = 10) and Listing (with id = 123). Then, delete the Listing record if there is no more association with any other Listing after this association deletion.

Slot

S/N	Type	End Points	Description
P	POST	/slots/listing/123	Create a new slot by using the request body data. Should return back the slot object (with id value). Do association of listing to slots here
P	GET	/slots/10	Get slots with id = 10
P	GET	/slots	Get all slots
	GET	/slots/query?startDate=	Get all slots with startDate
	GET	/slots/query?endDate=	Get all slots with endDate
	GET	/slots/query?startTime=	Get all slots with startTime
	GET	/slots/query?endTime=	Get all slots with endTime
	GET	/slots/query?avaliabilityEnum=	Get all slots with avaliabilityEnum
	GET	/slots/query?entityStatusEnum=	Get all slots with entityStatusEnum
P	PUT	/slots/10	Edit slot with id = 10 using the request body data (only basic attributes: startDate, endDate, startTime, endTime EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.
P	DELETE	/slots/10	Delete slot with id = 10
	POST	/slots/10/listing	Assign a listing who gave the feedback (using the request body data) with id = 10. Should return back the customer object (with id value).
	POST	/slots/10/booking	Assign a booking who gave the feedback (using the request body data) with id = 10. Should return back the customer object (with id value).
P	DELETE	/slots/10/listing/123	Delete the association between location (with id = 10) and Listing

			(with id = 123). Then, delete the Listing record if there is no more association with any other Listing after this association deletion.
P	DELETE	/slots/10/booking/123	Delete the association between slot (with id = 10) and Booking (with id = 123). Then, delete the Booking record if there is no more association with any other Booking after this association deletion.

Staff

S/N	Type	End Points	Description
P	POST	/staffs	Create a new staff by using the request body data. Should return back the staff object (with id value).
P	GET	/staffs/10	Get staff with id = 10
P	GET	/staffs	Get all staff
P	GET	/staffs/query?email=	Get all staffs with email
P	GET	/staffs/query?staffAccessRightEnum=	Get all staffs with department
P	GET	/staffs/query?email=___&password=____	Login with email and hashed password
P	POST	/staffs/login	Login with email and hashed password
P	PUT	/staffs/10	Edit staff with id = 10 using the request body data (only basic attributes: email, password, staffAccessRightEnum, EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.
P	DELETE	/staffs/10	Delete staff with id = 10

Transaction

Type	End Points	Description	Normal	Validate
POST	/customer/10/booking/10	Create a new transaction by using the request body data. Should return back the slot object (with id value).	O	
GET	/transactions/10	Get transaction with id = 10	O	
GET	/transactions	Get all transactions	O	
GET	/transactions/query?transactionDate=	Get all transactions with transactionDate	X May need to change the dates	
GET	/transactions/query?totalAmount=	Get all transactions with totalAmount	x	
GET	/transactions/query?transactionTypeEnum=	Get all transactions with transactionTypeEnum	x	
GET	/transactions/query?entityStatusEnum=	Get all transactions with entityStatusEnum	x	
PUT	/transactions/10	Edit transaction with id = 10 using the request body data (only basic attributes: transactionDate, totalAmount, transactionTypeEnum, EntityStatusEnum). Should return empty payload if the update is success or 404 if the customer is not found.	o	
DELETE	/transactions/10	Delete transaction with id = 10	o	
DELETE	/transactions/10/booking/123	Delete the association between transaction (with id = 10) and booking (with id = 123). Then, delete the		

		transaction record if there is no more association with any other booking after this association deletion.		

Comments