Parcels Tracking App

Parcels Tracking App is an application which can help hotel receptionists to trace, search and handle parcels for guests.

This application is Java REST based service which can receive, send, and handle data as http requests and responses messages in JSON format. Service was built using Maven tool, Lombok library, tests supported by Junit and Mockito. Data is stored using H2 embedded database, logging is handled by Log4j2 library.

Database using Hybernate Spring tool creates table PARCELS with columns:
ID, varchar(256), Primary Key, not null;
PARCEL_ID, varchar(256), not null;
CLIENT_ID, varchar(256);
PARCEL_TYPE, varchar(256);
PARCEL_REGISTERED, Date;
PARCEL_DELIVERED, Boolean;
PARCEL_DELIVERY_DATE, Date.

Main class which creates table is *ParcelEntity* with 7 fields:
Id - field with type String, cannot be null.
parcelId – field with type String, cannot be null.
clientId – field with type String, cannot be null.
parcelType – field with type String, cannot be null.
registrationDate – field with type LocalDate, can be null.
delivered – field with type Boolean, can be null.
deliveryDate – field with type String, can be null.

Parcels Tracking App implements Model View Controller design pattern.
Service has controller with the name *ParcelsTrackingAppController*, which creates 5 REST API's:

1. /newParcel;
2. /allParcels;
3. /parcelsByClientId;
4. /parcelById;
5. /updateParcel;

Application with the main address http://localhost:8800 handles http input and output requests:

1. **/newParcel** - API to register and store new parcel.
   Requests in JSON format with 6 fields are provided and validated by *StoreNewParcelRequest* which has 6 fields:

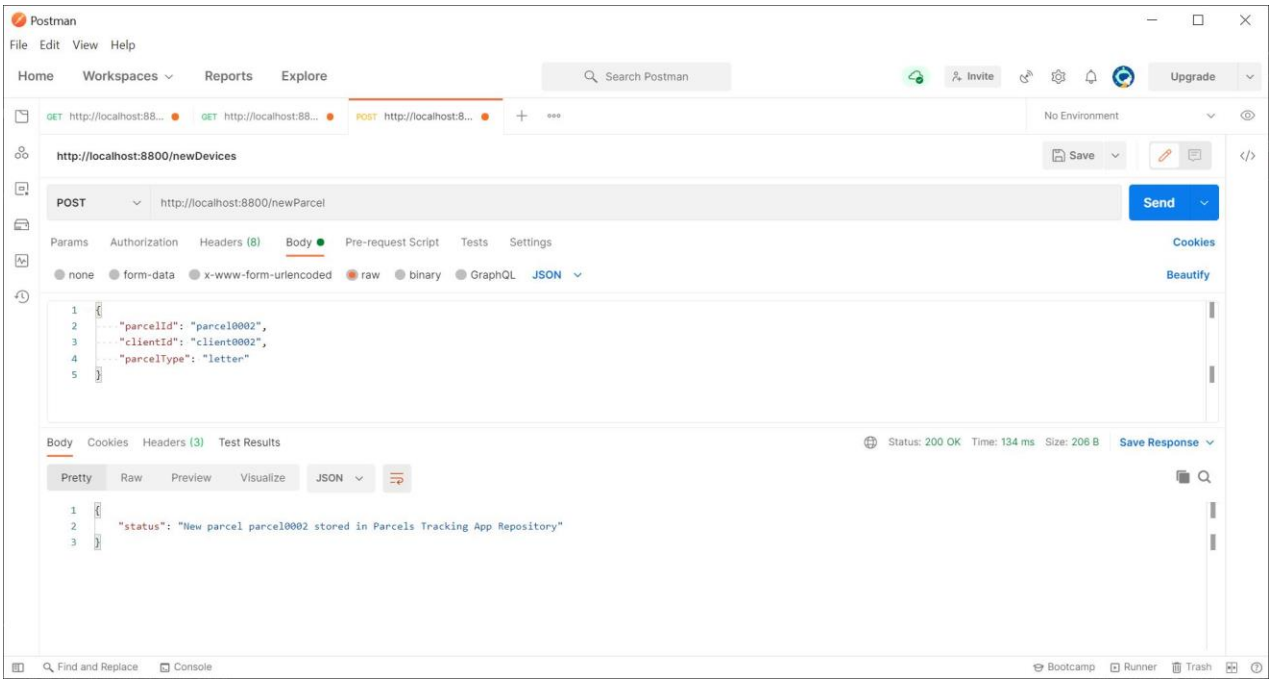
1.1. parcelId – field with type String, provided with @NotBlank annotation, cannot be null.
1.2. clientId – field with type String, provided with @NotBlank annotation, cannot be null.
1.3. parcelType – field with type String, provided with @NotBlank annotation, cannot be null.
1.4. registrationDate – field with type String, can be null.

1.5. delivered – field with type Boolean, can be null.
1.6. deliveryDate – field with type String, can be null.

Responses in JSON format with 1 field are provided by *StoreNewParcelResponse* which has 1 field:

1.7. status – field with type String, generated by *StoreNewParcelService*, cannot be null.

StoreNewParcelRequest and *StoreNewParcelResponse* are handled by *StoreNewParcelService*. Service provides validation for *StoreNewParcelRequest* fields by null and empty fields validation. *RequestValidator* additionally validates parcel delivery fields Boolean Delivered and String DeliveryDate.



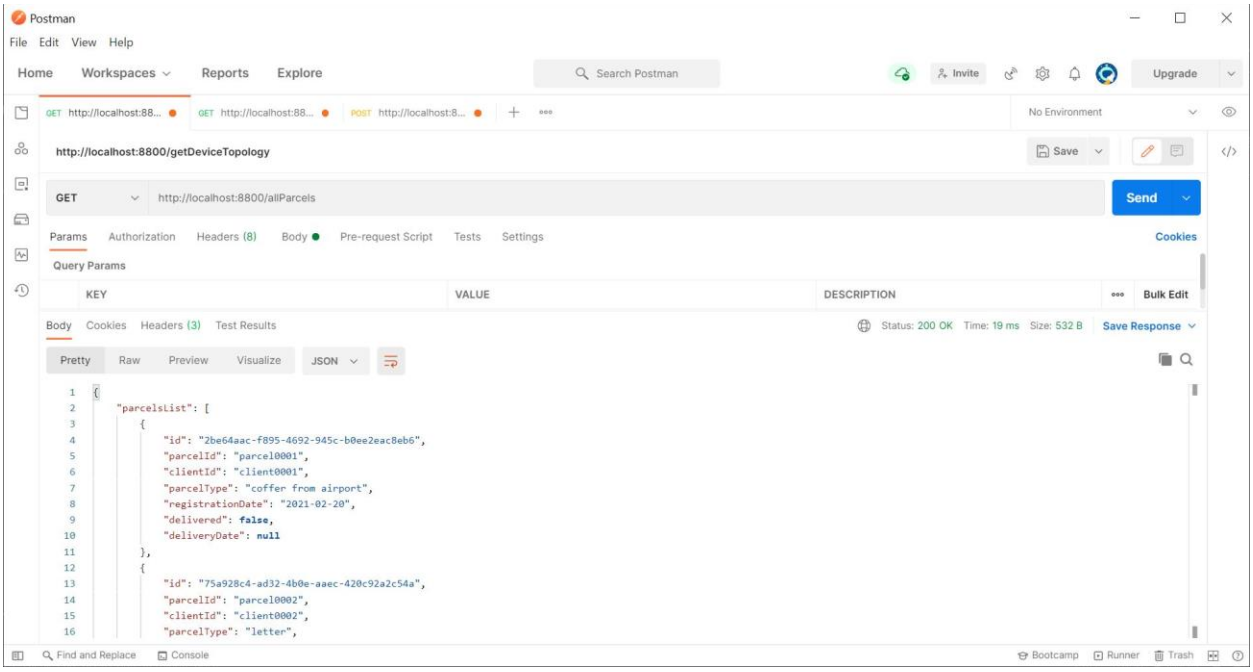
Picture 1. Store new parcel request and response.

2. **/allParcels** – API to get all parcels from Repository.
Requests with empty body are handled by *GetAllParcelsService.*

Responses in JSON format with 1 field are provided by *GetAllParcelsResponse* which has 1 field:

2.1. parcelsList – List with nodes by type *ParcelEntity*, generated by *GetAllParcelsService*, cannot be null.

Request and *GetAllParcelsResponse* are handled by *GetAllParcelsService*. Service provides validation for response form repository.



Picture 2. Get all parcels request and response.

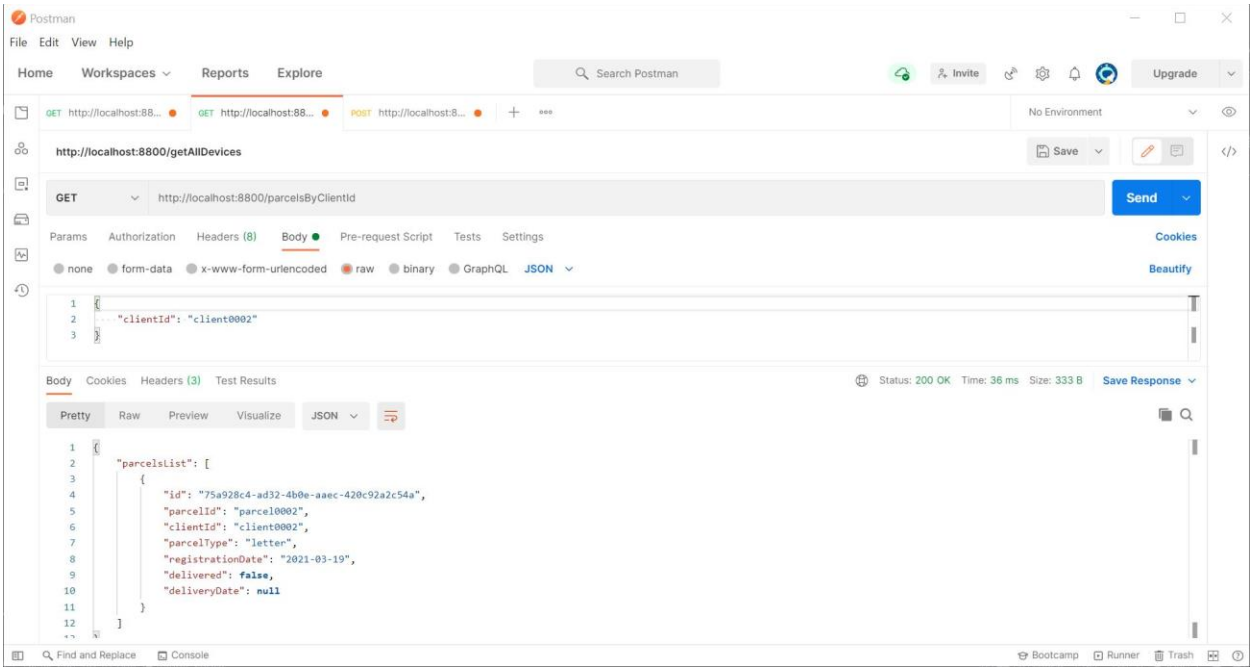
3. **/parcelsByClientId** – API get list of parcels by Client ID.
   Requests in JSON format with 1 field are provided and validated by *GetParcelsByClientIdRequest* which has 1 field:

3.1. clientId – field with type String, provided with @NotBlank annotation, cannot be null.

Responses in JSON format with 1 field are provided by *GetParcelsByClientIdResponse* which has 1 field:

3.2. parcelsList – List with nodes by type *ParcelEntity*, generated by *GetParcelsByClientIdService*, cannot be null.

GetParcelsByClientIdRequest and *GetParcelsByClientIdResponse* are handled by *GetParcelsByClientIdService*. Service provides validation for request as well as response form repository.



Picture 2. Get parcels by Client ID request and response.

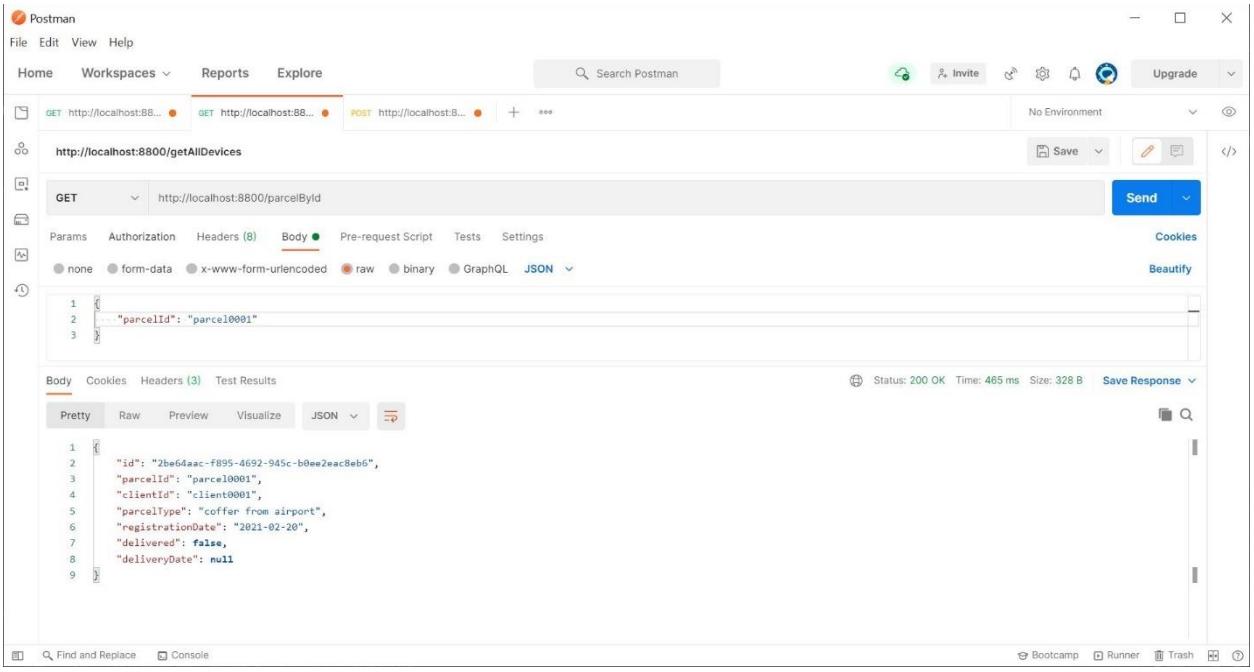
4. **/parcelById**– API get parcel by Parcel ID.
   Requests in JSON format with 1 field are provided and validated by *GetByParcelIdRequest* which has 1 field:

4.1. parcelId – field with type String, provided with @NotBlank annotation, cannot be null.

Responses in JSON format with 1 field are provided by *GetByParcelIdResponse* which has 7 field:

4.2. id – field with type String, generated by *GetByParcelIdService*, cannot be null.
4.3. parcelId – field with type String, cannot be null.
4.4. clientId – field with type String, cannot be null.
4.5. parcelType – field with type String, cannot be null.
4.6. registrationDate – field with type String, cannot be null.
4.7. delivered – field with type Boolean, cannot be null.
4.8. deliveryDate – field with type String, can be null.

GetByParcelIdRequest and *GetByParcelIdResponse* are handled by *GetByParcelIdService*. Service provides building of *GetByParcelIdResponse.*



Picture 4. Get parcel by Parcel ID request and response.

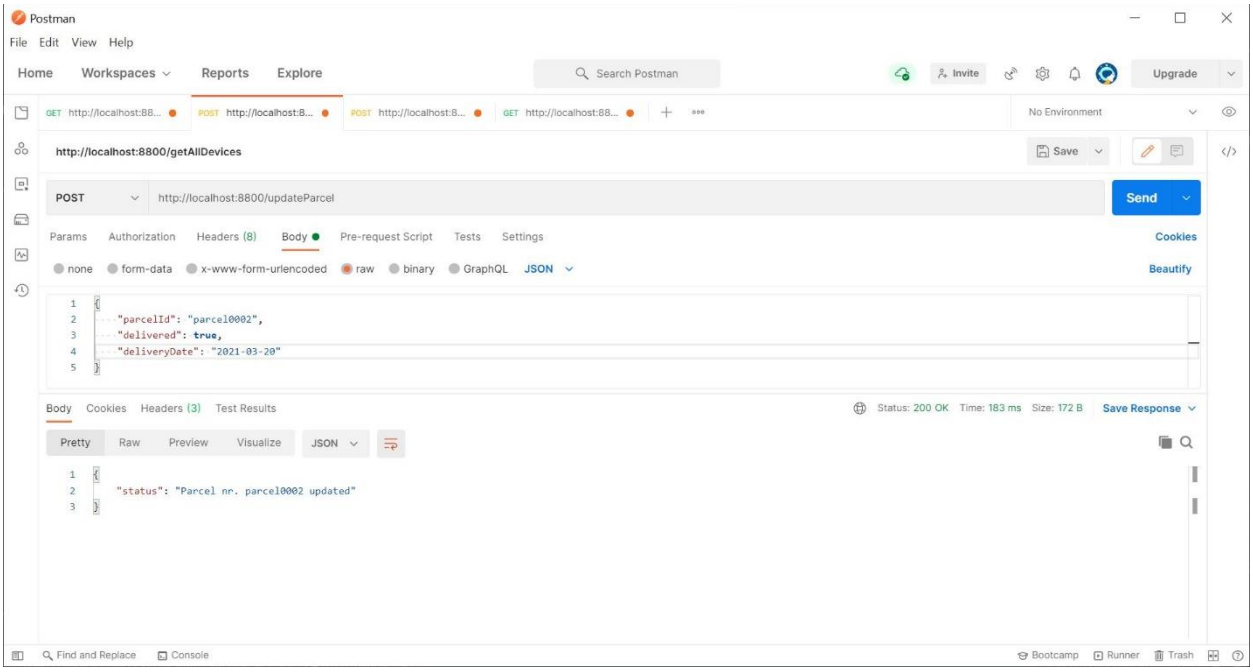
5. **/updateParcel** - API to update parcel.
   Requests in JSON format with 6 fields are provided and validated by *UpdateParcelRequest* which has 3 fields:

5.1. parcelId – field with type String, provided with @NotBlank annotation, cannot be null.
5.2. delivered – field with type Boolean, can be null.
5.3. deliveryDate – field with type String, can be null.

Responses in JSON format with 1 field are provided by *UpdateParcelResponse* which has 1 field:

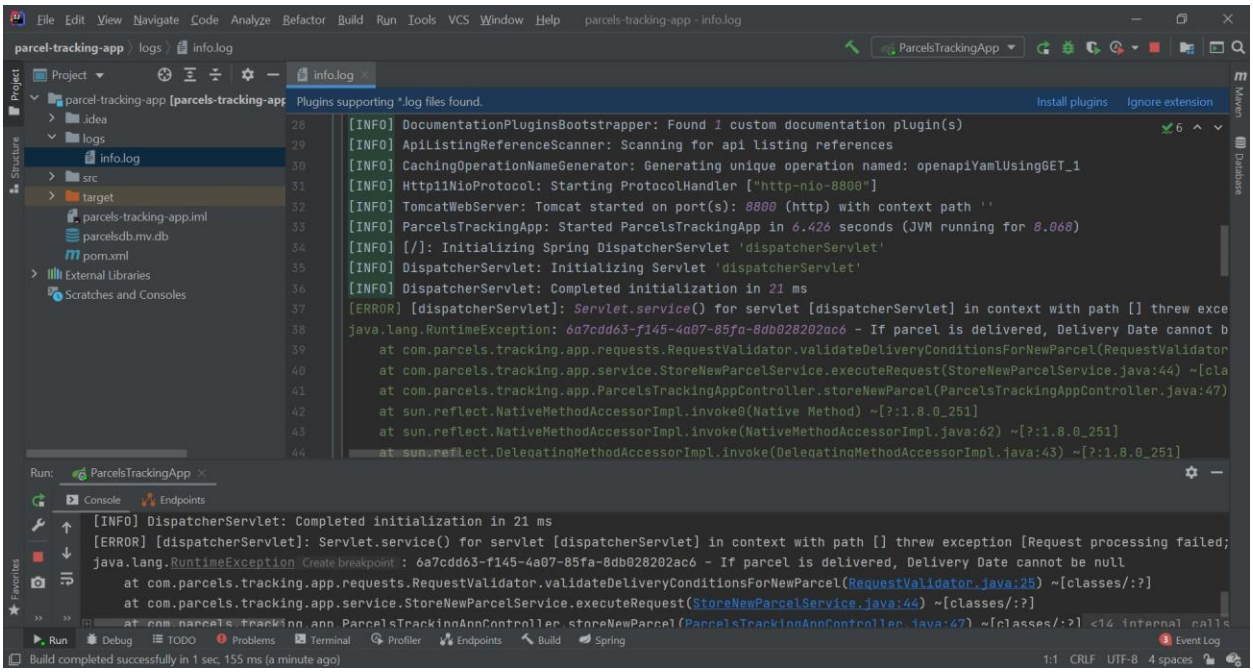
5.4. status – field with type String, generated by *UpdateParcelService*, cannot be null.

UpdateParcelRequest and *UpdateParcelResponse* are handled by *UpdateParcelService*. Service provides validation for request fields by null and empty fields validation.
RequestValidator additionally validates parcel delivery fields Boolean Delivered and String DeliveryDate.



Picture 5. Update parcel request and response.

Error, Warning and Info messages are handled by Log4j2 logging tool.



Picture 6. Log file.

Document created by Romans Jefremovs
Java Application Developer
email: rjefremovs@gmail.com