

TESTING DOCUMENT

Transporter:

1] Invalid data:

1. Without company name Request

Throws a custom Exception

Request and Response is shown in the below image

The screenshot shows the Postman interface with a request and response. The request body is a JSON object with an empty 'companyName' field. The response body is a JSON object containing a timestamp, status code 400, error message 'Bad Request', and a detailed message about an empty company name.

```
1 {  
2   "companyName": "",  
3   "rating": 4.5,  
4   "availableTrucks": [  
5     {  
6       "truckType": "Flatbed",  
7       "count": 5  
8     },  
9     {  
10      "truckType": "Refrigerated",  
11      "count": 3  
12    }  
13  ]  
14 }
```

```
1 {  
2   ... "timestamp": "2025-12-04T16:04:32.930750",  
3   ... "status": 400,  
4   ... "error": "Bad Request",  
5   ... "message": "companyName : Company name cannot be empty!",  
6   ... "path": "/transporter/"  
7 }
```

2. Rating out of boundary conditions

Both negative and rating over 5 is shown in below images Both conditions request and Response is shown below.

The screenshot shows the Postman interface with a request and response. The request body has a rating of 6. The response body is a JSON object containing a timestamp, status code 400, error message 'Bad Request', and a detailed message about a rating of 6.

```
1 {  
2   "companyName": "ABC Logistics",  
3   "rating": 6,  
4   "availableTrucks": [  
5     {  
6       "truckType": "Flatbed",  
7       "count": 5  
8     },  
9     {  
10      "truckType": "Refrigerated",  
11      "count": 3  
12    }  
13  ]  
14 }
```

The screenshot shows the Postman interface with a request and response. The request body has a rating of -1. The response body is a JSON object containing a timestamp, status code 400, error message 'Bad Request', and a detailed message about a rating of -1.

```
1 {  
2   "companyName": "ABC Logistics",  
3   "rating": -1,  
4   "availableTrucks": [  
5     {  
6       "truckType": "Flatbed",  
7       "count": 5  
8     },  
9     {  
10      "truckType": "Refrigerated",  
11      "count": 3  
12    }  
13  ]  
14 }
```

The screenshot shows the Postman interface with a request and response. The request body has a rating of 6. The response body is a JSON object containing a timestamp, status code 400, error message 'Bad Request', and a detailed message about a rating of 6.

```
1 {  
2   ... "timestamp": "2025-12-04T16:07:35.059147300",  
3   ... "status": 400,  
4   ... "error": "Bad Request",  
5   ... "message": "rating : Rating must be at most 5",  
6   ... "path": "/transporter/"  
7 }
```

The screenshot shows the Postman interface with a request and response. The request body has a rating of -1. The response body is a JSON object containing a timestamp, status code 400, error message 'Bad Request', and a detailed message about a rating of -1.

```
1 {  
2   ... "timestamp": "2025-12-04T16:06:41.557172300",  
3   ... "status": 400,  
4   ... "error": "Bad Request",  
5   ... "message": "rating : Rating must be at least 0",  
6   ... "path": "/transporter/"  
7 }
```

Create Transporter:

Post method → URL: <http://localhost:8080/transporter/>

2] Valid data: creating Transporter1 (1st company)

Request:

```
1 {  
2   "companyName": "ABC Logistics",  
3   "rating": 4.5,  
4   "availableTrucks": [  
5     {  
6       "truckType": "Flatbed",  
7       "count": 5  
8     },  
9     {  
10       "truckType": "Refrigerated",  
11       "count": 3  
12     }  
13   ]  
14 }
```

Response:

```
1 {  
2   "message": "transporter created successfully!",  
3   "data": {  
4     "transporterId": "9d0f4031-9781-4084-b901-d6d2bdfd0057",  
5     "companyName": "ABC Logistics",  
6     "rating": 4.5,  
7     "availableTrucks": [  
8       {  
9         "id": 1,  
10        "truckType": "Flatbed",  
11        "count": 5  
12      },  
13      {  
14        "id": 2,  
15        "truckType": "Refrigerated",  
16        "count": 3  
17      }  
18    ],  
19    "status": "CREATED"  
20  }  
21 }
```

creating Transporter2 as 2nd Company without truck Details

Post method → URL: <http://localhost:8080/transporter/>

```
1 {  
2   "companyName": "XYZ Logistics",  
3   "rating": 4.5  
4 }
```

Body Cookies Headers (5) Test Results ⚙

201 Created • 34 ms • 365 B ⓘ

{ } JSON ▾ ▷ Preview ⌂ Visualize ▾

```
1 {  
2   "message": "transporter created successfully!",  
3   "data": {  
4     "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
5     "companyName": "XYZ Logistics",  
6     "rating": 4.5,  
7     "availableTrucks": null  
8   },  
9   "status": "CREATED"  
10 }
```

UpdatAvailableTrucks:

Request Method: PUT

URL: <http://localhost:8080/transporter/{mention here transporterId} /trucks>

Updating truck details to the transporter 2 (2nd company)

Request:

```
1 [  
2 {  
3   "truckType": "Flatbed",  
4   "count": 5  
5 },  
6 {  
7   "truckType": "Refrigerated",  
8   "count": 3  
9 }  
10 ]
```

Response:

```
1 {  
2   "message": "Available trucks updated successfully!",  
3   "data": {  
4     "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
5     "companyName": "XYZ Logistics",  
6     "rating": 4.5,  
7     "availableTrucks": [  
8       {  
9         "id": 3,  
10        "truckType": "Flatbed",  
11        "count": 5  
12      },  
13      {  
14        "id": 4,  
15        "truckType": "Refrigerated",  
16        "count": 3  
17      }  
18    ],  
19  },  
20  "status": "UPDATED"  
21 }
```

Updating Existing Trucks count:

Using GetTransporter check the transporter details first.

Request and Response: Before updating the transporter.

Body Cookies Headers (5) Test Results ⌂

{ } JSON ▾ ▷ Preview 🖼 Visualize ▾

```
1 {  
2   "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
3   "companyName": "ABC Logistics",  
4   "rating": 4.5,  
5   "availableTrucks": [  
6     {  
7       "id": 52,  
8       "truckType": "Flatbed",  
9       "count": 5  
10    },  
11    {  
12      "id": 53,  
13      "truckType": "Refrigerated",  
14      "count": 3  
15    }  
16  ]  
17 }
```

Then update specific truck count and check

Request and Response:

Request and Response show below after updating.

```
1 [  
2   {  
3     "truckType": "Flatbed",  
4     "count": 3  
5   }  
6 ]
```

Body Cookies Headers (5) Test Results | ⏱

{ } JSON ▾ ▷ Preview Visualize | ▾

```
1 {  
2   "message": "Available trucks updated successfully!",  
3   "data": {  
4     "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
5     "companyName": "ABC Logistics",  
6     "rating": 4.5,  
7     "availableTrucks": [  
8       {  
9         "id": 52,  
10        "truckType": "Flatbed",  
11        "count": 8  
12      },  
13      {  
14        "id": 53,  
15        "truckType": "Refrigerated",  
16        "count": 3  
17      }  
18    ]  
19  },  
20  "status": "UPDATED"  
21 }
```

Creating 3rd Transporter as Transporter3 (3rd company) Post method → URL: <http://localhost:8080/transporter/>

Request:

```
1 {  
2   "companyName": "MNC Logistics",  
3   "rating": 4.5,  
4   "availableTrucks": [  
5     {  
6       "truckType": "Flatbed",  
7       "count": 1  
8     },  
9     {  
10       "truckType": "Refrigerated",  
11       "count": 1  
12     }  
13   ]  
14 }
```

Response:

```
1 {  
2   "message": "transporter created successfully!",  
3   "data": {  
4     "transporterId": "3af4437c-617a-4df6-800a-99852b9cf1de",  
5     "companyName": "MNC Logistics",  
6     "rating": 4.5,  
7     "availableTrucks": [  
8       {  
9         "id": 5,  
10        "truckType": "Flatbed",  
11        "count": 1  
12      },  
13      {  
14        "id": 6,  
15        "truckType": "Refrigerated",  
16        "count": 1  
17      }  
18    ]  
19  },  
20  "status": "CREATED"  
21 }
```

GetTransporter: getting the transporter using transporter id

Request method: "GET"

URL: <http://localhost:8080/transporter/{transporterId}>

Response:

```
{ } JSON ▾ ▷ Preview ⚡ Visualize | ▾  
1 {  
2   "transporterId": "3af4437c-617a-4df6-800a-99852b9cf1de",  
3   "companyName": "MNC Logistics",  
4   "rating": 4.5,  
5   "availableTrucks": [  
6     {  
7       "id": 5,  
8       "truckType": "Flatbed",  
9       "count": 1  
10    },  
11    {  
12      "id": 6,  
13      "truckType": "Refrigerated",  
14      "count": 1  
15    }  
16  ]  
17 }
```

Load:

Request Method: POST

URL: <http://localhost:8080/load/>

1] Invalid Data:

1. Invalid Shipper Id: Try to create load with blank shipperId or empty.or don't pass shipperId
Request and Response is shown below

```
1 {  
2   "shipperId": "",  
3   "loadingCity": "Delhi",  
4   "unloadingCity": "Mumbai",  
5   "loadingDate": "2025-12-05T10:00:00",  
6   "productType": "Electronics",  
7   "weight": 1000,  
8   "weightUnit": "KG",  
9   "truckType": "Flatbed",  
10  "noOfTrucks": 2  
11 }
```

Body Cookies Headers (4) Test Results | ⚡

```
{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾  
1 {  
2   ... "timestamp": "2025-12-04T17:04:37.026179100",  
3   ... "status": 400,  
4   ... "error": "Bad Request",  
5   ... "message": "shipperId : shipper id cannot be empty!",  
6   ... "path": "/load/"  
7 }
```

2. Request without Loading city: without passing loading city try to create the load or keep the loading city blank or empty.

Request & Response is shown below

```

1  {
2    "shipperId": "shipper123",
3    "loadingCity": "",
4    "unloadingCity": "Mumbai",
5    "loadingDate": "2025-12-05T10:00:00",
6    "productType": "Electronics",
7    "weight": 1000,
8    "weightUnit": "KG",
9    "truckType": "Flatbed",
10   "noOfTrucks": 2
11 }
```

Body Cookies Headers (4) Test Results | ⚙️

{ } JSON ▾ ▶ Preview ⚙️ Debug with AI | ▾

```

1  {
2    ... "timestamp": "2025-12-04T17:05:25.151126100",
3    ... "status": 400,
4    ... "error": "Bad Request",
5    ... "message": "loadingCity : Loading city cannot be empty!",
6    ... "path": "/load/"
7 }
```

3. Request without unloading city: Try to create Load without unloading city. Or keep unloading city blank or empty.

Request and Response :

```

1  {
2    "shipperId": "shipper123",
3    "loadingCity": "Delhi",
4    "unloadingCity": "",
5    "loadingDate": "2025-12-05T10:00:00",
6    "productType": "Electronics",
7    "weight": 1000,
8    "weightUnit": "KG",
9    "truckType": "Flatbed",
10   "noOfTrucks": 2
11 }
```

Body Cookies Headers (4) Test Results | ⚙️

{ } JSON ▾ ▶ Preview ⚙️ Debug with AI | ▾

```

1  {
2    ... "timestamp": "2025-12-04T17:06:21.124721700",
3    ... "status": 400,
4    ... "error": "Bad Request",
5    ... "message": "unloadingCity : Unloading city cannot be empty!",
6    ... "path": "/load/"
7 }
```

4. Request with Passed date: try to create Load with passed date as loading time

Request and Response:

```
1 {  
2   "shipperId": "shipper123",  
3   "loadingCity": "Delhi",  
4   "unloadingCity": "Mumbai",  
5   "loadingDate": "2023-12-01T10:00:00",  
6   "productType": "Electronics",  
7   "weight": 1000,  
8   "weightUnit": "KG",  
9   "truckType": "Flatbed",  
10  "noOfTrucks": 2  
11 }
```

Body Cookies Headers (4) Test Results | ⓘ

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```
1 {  
2   "timestamp": "2025-12-04T16:18:09.993930700",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "Loading date is already passed!",  
6   "path": "/load/"  
7 }
```

5. Invalid weight: try to create with invalid weight like negative

Request and Response:

```
1 {  
2   "shipperId": "shipper123",  
3   "loadingCity": "Delhi",  
4   "unloadingCity": "Mumbai",  
5   "loadingDate": "2025-12-05T10:00:00",  
6   "productType": "Electronics",  
7   "weight": -1000,  
8   "weightUnit": "KG",  
9   "truckType": "Flatbed",  
10  "noOfTrucks": 2  
11 }
```

Body Cookies Headers (4) Test Results | ⓘ

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```
1 {  
2   "timestamp": "2025-12-04T17:08:36.999127200",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "weight :: must be greater than or equal to 0",  
6   "path": "/load/"  
7 }
```

6. Invalid weight Unit: try to create a load with invalid weight unit.

Request and Response

```
1 {  
2   "shipperId": "shipper123",  
3   "loadingCity": "Delhi",  
4   "unloadingCity": "Mumbai",  
5   "loadingDate": "2025-12-06T10:00:00",  
6   "productType": "Electronics",  
7   "weight": 1000,  
8   "weightUnit": "MG",  
9   "truckType": "Flatbed",  
10  "noOfTrucks": 2  
11 }
```

Body Cookies Headers (4) Test Results | ⓘ

400 Bad Request • 18 ms • 412 B • ⓘ Save Res

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```
1 {  
2   "timestamp": "2025-12-04T11:40:08.075+00:00",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "JSON parse error: Cannot deserialize value of type `com.tmc.model.load.WeightUnit` from String \"MG\": not one of the values accepted for Enum class: [TON, KG]",  
6   "path": "/load/"  
7 }
```

7. Invalid noOfTrucks: try creating load with invalid noOfTrucks like negative number.

Request & Response

```

1  {
2    "shipperId": "shipper123",
3    "loadingCity": "Delhi",
4    "unloadingCity": "Mumbai",
5    "loadingDate": "2025-12-05T10:00:00",
6    "productType": "Electronics",
7    "weight": 1000,
8    "weightUnit": "KG",
9    "truckType": "Flatbed",
10   "noOfTrucks": -1
11 }

```

Body Cookies Headers (4) Test Results | ⚡

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```

1  {
2    ...
3    "timestamp": "2025-12-04T17:12:00.990729",
4    "status": 400,
5    "error": "Bad Request",
6    "message": "noOfTrucks : must be greater than or equal to 0",
7    "path": "/load/"
}

```

8. Invalid truckType: Create load with empty or blank truckType.

Request and Response

```

1  {
2    "shipperId": "shipper123",
3    "loadingCity": "Delhi",
4    "unloadingCity": "Mumbai",
5    "loadingDate": "2025-12-05T10:00:00",
6    "productType": "Electronics",
7    "weight": 1000,
8    "weightUnit": "KG",
9    "truckType": "",
10   "noOfTrucks": 2
11 }

```

Body Cookies Headers (4) Test Results | ⚡

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```

1  {
2    ...
3    "timestamp": "2025-12-04T17:11:28.461067400",
4    "status": 400,
5    "error": "Bad Request",
6    "message": "truckType : truckType cannot be empty!",
7    "path": "/load/"
}

```

2] Valid data: create load with valid data.

```

1  {
2    "shipperId": "shipper123",
3    "loadingCity": "Delhi",
4    "unloadingCity": "Mumbai",
5    "loadingDate": "2025-12-05T10:00:00",
6    "productType": "Electronics",
7    "weight": 1000,
8    "weightUnit": "KG",
9    "truckType": "Flatbed",
10   "noOfTrucks": 2
11 }

```

Body Cookies Headers (5) Test Results | ⚡

{ } JSON ▾ ▷ Preview ⚡ Visualize | ▾

```

1  {
2    "message": "Load created Successfully!",
3    "data": {
4      "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",
5      "shipperId": "shipper123",
6      "loadingCity": "Delhi",
7      "unloadingCity": "Mumbai",
8      "loadingDate": "2025-12-05T10:00:00",
9      "productType": "Electronics",
10     "weight": 1000.0,
11     "weightUnit": "KG",
12     "truckType": "Flatbed",
13     "noOfTrucks": 2,
14     "status": "POSTED",
15     "datePosted": "2025-12-04T16:19:40.113656"
16   },
17   "status": "CREATED"
18 }

```

getLoad:

RequestMethod: GET

URL: <http://localhost:8080/load/{loadId}>

Response:

```
{ } JSON ▾ ▷ Preview ⚖ Visualize | ▾  
1 {  
2   "loadId": "866af903-c74d-4e31-895f-c0d727065948",  
3   "shipperId": "shipper124",  
4   "loadingCity": "Delhi",  
5   "unloadingCity": "Mumbai",  
6   "loadingDate": "2025-12-10T10:00:00",  
7   "productType": "Electronics",  
8   "weight": 1000.0,  
9   "weightUnit": "KG",  
10  "truckType": "Flatbed",  
11  "noOfTrucks": 1,  
12  "status": "CANCELLED",  
13  "datePosted": "2025-12-04T16:21:22.016162"  
14 }
```

cancelLoad:

RequestMethod: PATCH

URL: <http://localhost:8080/load/{loadId} /cancel>

Response:

```
{ } JSON ▾ ▷ Preview ⚖ Visualize | ▾  
1 {  
2   "message": "Load cancelled successfully!",  
3   "data": {  
4     "loadId": "866af903-c74d-4e31-895f-c0d727065948",  
5     "shipperId": "shipper124",  
6     "loadingCity": "Delhi",  
7     "unloadingCity": "Mumbai",  
8     "loadingDate": "2025-12-10T10:00:00",  
9     "productType": "Electronics",  
10    "weight": 1000.0,  
11    "weightUnit": "KG",  
12    "truckType": "Flatbed",  
13    "noOfTrucks": 1,  
14    "status": "CANCELLED",  
15    "datePosted": "2025-12-04T16:21:22.016162"  
16  },  
17  "status": "UPDATED"  
18 }
```

cancelLoad:

RequestMethod: GET

URL: <http://localhost:8080/load/{loadId} /best-bids>

Response:

```
1 [  
2   {  
3     "bidId": "7f849798-e710-4686-992f-957a270b53cc",  
4     "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
5     "proposedRate": 4000.0,  
6     "rating": 4.5,  
7     "score": 0.270175  
8   },  
9   {  
10     "bidId": "a26dd757-c4b5-4afe-b4de-061f37a883c2",  
11     "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
12     "proposedRate": 5000.0,  
13     "rating": 4.5,  
14     "score": 0.27014  
15   }  
16 ]
```

If bids are not present for particular load then.

Response:

```
Body Cookies Headers (5) Test Results | ⚙️
{ } JSON ▾ ▶ Preview ⚡ Debug with AI | ▾
1  {
2    "timestamp": "2025-12-04T17:29:01.970044200",
3    "status": 404,
4    "error": "Not Found",
5    "message": "Best-bids not found!",
6    "path": "/load/15d60b56-2084-4920-a123-57a6bfcf76bf/best-bids"
7 }
```

GetLoadsPaginated: Filtered on any one parameter either shipperId or status.

RequestMethod: GET

URL: <http://localhost:8080/load/>

Request and Response:

```
1  {
2    "shipperId": "SHIP778899",
3    "status": "POSTED",
4    "page": "0",
5    "size": "1"
6 }

Body Cookies Headers (5) Test Results | ⚙️
{ } JSON ▾ ▶ Preview ⚡ Visualize | ▾
1  {
2    "content": [
3      {
4        "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",
5        "shipperId": "shipper123",
6        "loadingCity": "Delhi",
7        "unloadingCity": "Mumbai",
8        "loadingDate": "2025-12-05T10:00:00",
9        "productType": "Electronics",
10       "weight": 1000.0,
11       "weightUnit": "KG",
12       "truckType": "Flatbed",
13       "noOfTrucks": 2,
14       "status": "POSTED",
15       "datePosted": "2025-12-04T16:19:40.113656",
16       "version": 0
17     }
18   ],
19   "pageable": {
20     "pageNumber": 0,
21     "pageSize": 1,
22     "sort": {
23       "empty": true,
24       "sorted": false,
25       "unsorted": true
26     },
27     "offset": 0,
28     "paged": true,
```

Bid:

CreateBid

RequestMethod: POST

URL: <http://localhost:8080/bid/>

1] Invalid data:

1. InsufficientTrucksOffered: try creating bid with trucks offered more than available trucks of transporter.

Request and Response:

```
1 {  
2   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
3   "transporterId": "3af4437c-617a-4df6-800a-99852b9cf1de",  
4   "proposedRate": 4000,  
5   "trucksOffered": 10  
6 }
```

Body Cookies Headers (4) Test Results | ⚙️

{ } JSON ▾ ▷ Preview ⚡ Debug with AI | ▾

```
1 {  
2   "timestamp": "2025-12-04T17:34:30.966322300",  
3   "status": 400,  
4   "error": "Bad Request",  
5   "message": "Not enough trucks of this type!",  
6   "path": "/bid/"  
7 }
```

Valid data: create bid for one load

Bid1:

Request and Response:

```
1 {  
2   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
3   "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
4   "proposedRate": 5000,  
5   "trucksOffered": 2  
6 }
```

Body Cookies Headers (5) Test Results | ⚙️

{ } JSON ▾ ▷ Preview 🖥 Visualize | ▾

```
1 {  
2   "bidId": "a26dd757-c4b5-4afe-b4de-061f37a883c2",  
3   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
4   "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
5   "proposedRate": 5000.0,  
6   "trucksOffered": 2,  
7   "status": "PENDING",  
8   "submittedAt": "2025-12-04T17:32:45.515277"  
9 }
```

Bid2:

Create 2nd bid for same load

Request and Response:

```
1 {  
2   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
3   "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
4   "proposedRate": 4000,  
5   "trucksOffered": 1  
6 }
```

Body Cookies Headers (5) Test Results | ⌂

{ } JSON ▾ ▷ Preview ⌂ Visualize | ▾

```
1 {  
2   "bidId": "7f849798-e710-4686-992f-957a270b53cc",  
3   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
4   "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
5   "proposedRate": 4000.0,  
6   "trucksOffered": 1,  
7   "status": "PENDING",  
8   "submittedAt": "2025-12-04T17:50:23.378148"  
9 }
```

getBid:

RequestMethod: GET

URL: <http://localhost:8080/bid/{bidId}>

Response:

{ } JSON ▾ ▷ Preview ⌂ Visualize | ▾

```
1 {  
2   "bidId": "a26dd757-c4b5-4afe-b4de-061f37a883c2",  
3   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
4   "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
5   "proposedRate": 5000.0,  
6   "trucksOffered": 2,  
7   "status": "PENDING",  
8   "submittedAt": "2025-12-04T17:32:45.515277"  
9 }
```

GetFilteredBids

RequestMethod: GET

URL: <http://localhost:8080/bid/>

Response:

```
1 {  
2   "loadId": "052cf7ac-d35a-410c-b199-45cafacc729c",  
3   "transporterId": "c946d43f-8eae-4714-b99d-3d464182f341",  
4   "status": "PENDING"  
5 }
```

Body Cookies Headers (5) Test Results | ⌂

{ } JSON ▾ ▷ Preview ⌂ Visualize | ▾

```
1 [  
2   {  
3     "bidId": "a26dd757-c4b5-4afe-b4de-061f37a883c2",  
4     "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
5     "transporterId": "f0ff8f74-810a-4563-b2b4-1c507755b16e",  
6     "proposedRate": 5000.0,  
7     "trucksOffered": 2,  
8     "status": "PENDING",  
9     "submittedAt": "2025-12-04T17:32:45.515277",  
10    "version": 0  
11  }  
12 ]
```

After checking best bids suggestions, we can choose which bid we want book and using that bidId we will book.

Booking

CreateBooking:

RequestMethod: POST

URL: <http://localhost:8080/booking/>

Request and Response

Booking1:

```
1  {
2    "bidId": "7f849798-e710-4686-992f-957a270b53cc"
3
4 }
```

Body Cookies Headers (5) Test Results |

{ } JSON ▾ ▷ Preview Visualize ▾

```
1  {
2    "message": "Booking created successfully!",
3    "data": [
4      {
5        "bookingId": "83155fa3-8f2a-4283-be2c-811b6a602031",
6        "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",
7        "bidId": "7f849798-e710-4686-992f-957a270b53cc",
8        "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",
9        "allocatedTrucks": 1,
10       "finalRate": 4000.0,
11       "status": "CONFIRMED",
12       "bookedAt": null
13     },
14   ],
15   "status": "CREATED"
16 }
```

getBooking:

RequestMethod: GET

URL: <http://localhost:8080/booking/bookingId>

Request and Response

{ } JSON ▾ ▷ Preview Visualize ▾

```
1  {
2    "bookingId": "83155fa3-8f2a-4283-be2c-811b6a602031",
3    "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",
4    "bidId": "7f849798-e710-4686-992f-957a270b53cc",
5    "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",
6    "allocatedTrucks": 1,
7    "finalRate": 4000.0,
8    "status": "CONFIRMED",
9    "bookedAt": "2025-12-04T17:55:11.831269"
10 }
```

cancelBooking:

RequestMethod: PATCH

URL: `http://localhost:8080/booking/bookingId/cancel`

Request and Response

```
PATCH http://localhost:8080/booking/83155fa3-8f2a-4283-be2c-811b6a602031/cancel

Docs Params Authorization Headers (8) Body Scripts Settings
Body Cookies Headers (5) Test Results | ⏱
{ } JSON ▾ ▷ Preview 🖼 Visualize | ▾
1 {  
2   "bookingId": "83155fa3-8f2a-4283-be2c-811b6a602031",  
3   "loadId": "65123255-490b-4f66-aaca-075f1b8af8f2",  
4   "bidId": "7f849798-e710-4686-992f-957a270b53cc",  
5   "transporterId": "4c857af5-cd39-45eb-9b28-7bfe2bb358ff",  
6   "allocatedTrucks": 1,  
7   "finalRate": 4000.0,  
8   "status": "CANCELLED",  
9   "bookedAt": "2025-12-04T17:55:11.831269"  
10 }
```