

Group Member Info:

Member 1:

Name: Sashank Malladi

Sjsu id: 010466651

Email: sashank.malladi@sjsu.edu

Member 2:

Name: prudhviraj mulagapati

Sjsu id: 010752495

Email: prudhviraj.mulagapati@sjsu.edu

.....

Cloud service Choice : Amazon

Hostname: ec2-52-26-65-149.us-west-2.compute.amazonaws.com

IP Address: 52.26.65.149

.....

## Screen Shots of the implementation

### Passenger Services

Fetch passenger using the passenger id. Example for URL is below

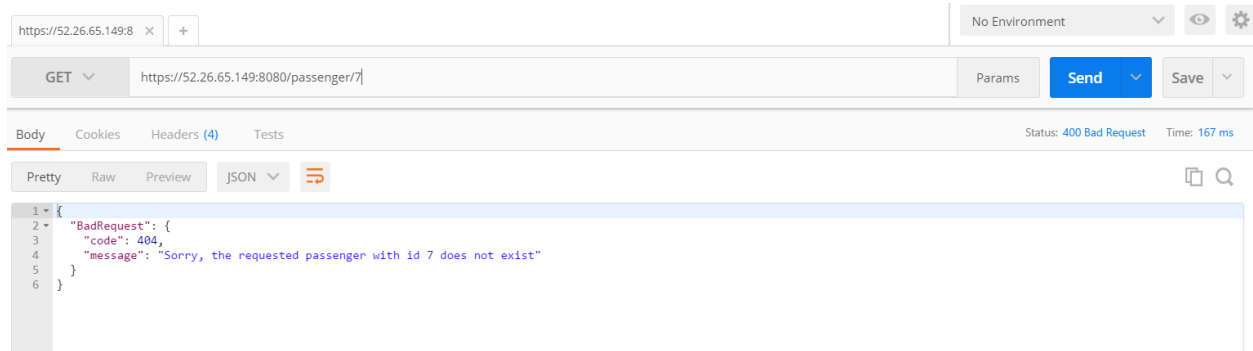
<https://52.26.65.149:8080/passenger/3>

The screenshot shows a REST client interface with a GET request to `https://52.26.65.149:8080/passenger/3`. The response is a JSON object representing a passenger and their reservations.

```
1 {
2   "passenger": {
3     "id": "3",
4     "firstname": "Prudhvi",
5     "lastname": "Raj",
6     "age": "26",
7     "gender": "male",
8     "phone": "992123",
9     "reservations": [
10      {
11        "orderNumber": "2",
12        "price": 250,
13        "flights": [
14          {
15            "price": "120",
16            "from": "Hyd",
17            "to": "Ban",
18            "departureTime": "2017-04-27-21",
19            "arrivalTime": "2017-04-27-23",
20            "seatsLeft": "99",
21            "description": "HyderabadtoBangalore",
22            "plane": {
23              "capacity": "100",
24              "model": "777",
25              "manufacturer": "Boeing",
26              "yearOfManufacture": "1997"
27            },
28            "number": "AA1"
29          },
30          {
31            "price": "130",
32            "from": "Hyd",
33            "to": "Del",
34            "departureTime": "2017-04-27-06",
35            "arrivalTime": "2017-04-27-09",
36            "seatsLeft": "99",
37            "description": "HyderabadtoDelhi",
38            "plane": {
39              "capacity": "100",
40              "model": "777",
41              "manufacturer": "Boeing",
42              "yearOfManufacture": "1997"
43            },
44            "number": "AA2"
45          }
46        ]
47      }
48    ]
49  }
50 }
```

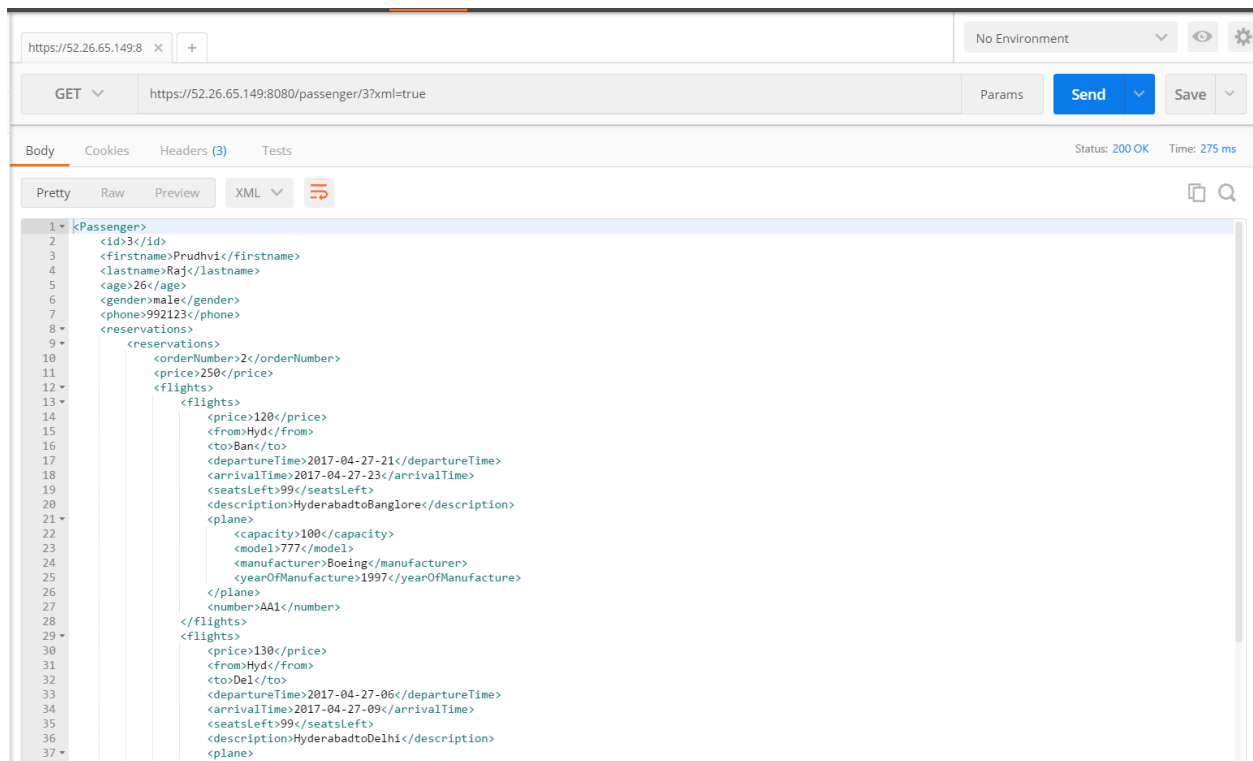
Fetch using passenger id. Example for URL is below

<https://52.26.65.149:8080/passenger/7>



Fetch passenger for xml . Example for URL is below

<https://52.26.65.149:8080/passenger/3?xml=true>



Fetch passenger for xml with bad request. Example for URL is below

<https://52.26.65.149:8080/passenger/7?xml=true>

The screenshot shows a REST client interface with a GET request to `https://52.26.65.149:8080/passenger/7?xml=true`. The response status is `400 Bad Request` with a time of `111 ms`. The response body is displayed in XML format:

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>404</code>
4     <message>Sorry, the requested passenger with id 7 does not exist</message>
5   </BadRequest>
6 </ExceptionResponse>
```

Create passenger. Example for URL is below

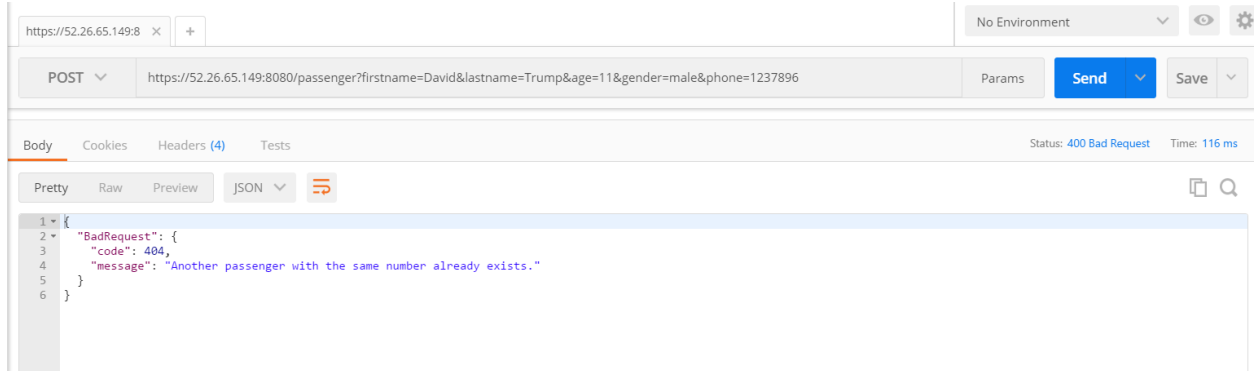
<https://52.26.65.149:8080/passenger?firstname=David&lastname=Trump&age=11&gender=male&phone=1237896>

The screenshot shows a REST client interface with a POST request to `https://52.26.65.149:8080/passenger?firstname=David&lastname=Trump&age=11&gender=male&phone=1237896`. The response status is `200 OK` with a time of `225 ms`. The response body is displayed in JSON format:

```
1 {
2   "passenger": {
3     "id": "5",
4     "firstname": "David",
5     "lastname": "Trump",
6     "age": "11",
7     "gender": "male",
8     "phone": "1237896",
9     "reservations": []
10  }
11 }
```

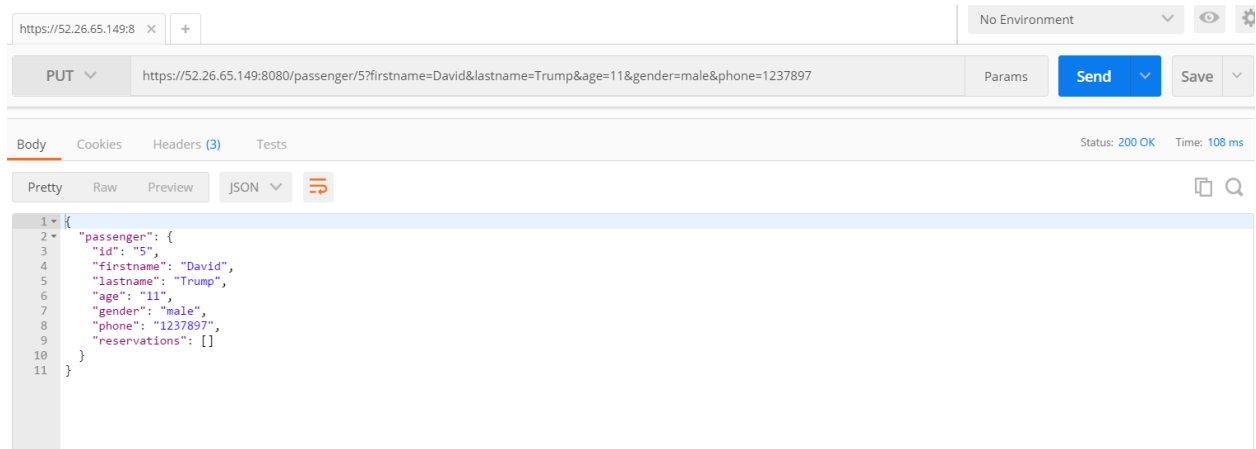
Create passenger for invalid request (Repeated Phone Number). Example for URL is below

<https://52.26.65.149:8080/passenger?firstname=David&lastname=Trump&age=11&gender=male&phone=1237896>



Update passenger. Example for URL is below

<https://52.26.65.149:8080/passenger/5?firstname=David&lastname=Trump&age=11&gender=male&phone=1237897>



Update passenger invalid request (Repeated phone number). Example for URL is below

https://52.26.65.149:8080/passenger/5?firstname=David&lastname=Trump&age=11&gender=male&phone=12345678

The screenshot shows a REST client interface with a PUT request to the URL `https://52.26.65.149:8080/passenger/5?firstname=David&lastname=Trump&age=11&gender=male&phone=12345678`. The response status is `400 Bad Request` with a time of `108 ms`. The response body is displayed in JSON format:

```
1 {
2   "BadRequest": {
3     "code": 404,
4     "message": "Another passenger with the same number already exists."
5   }
6 }
```

Delete Passenger. Example for URL is below

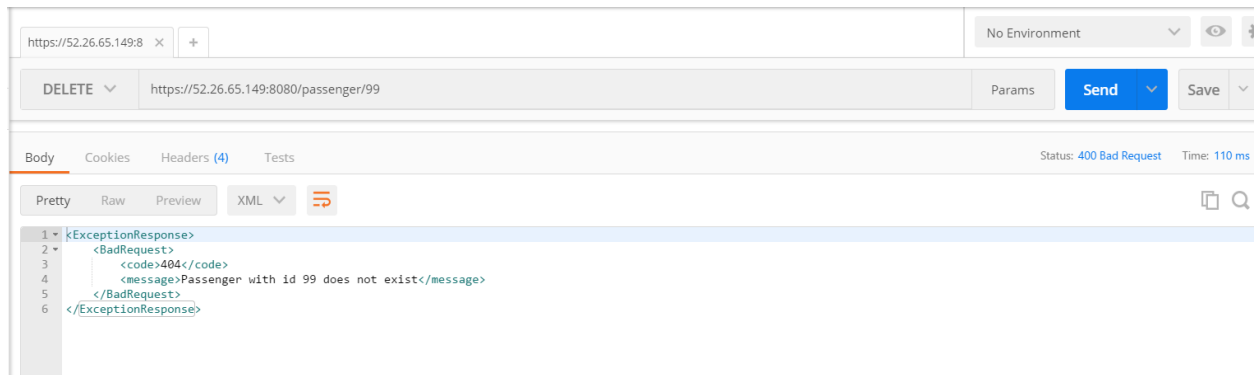
https://52.26.65.149:8080/passenger/5

The screenshot shows a REST client interface with a DELETE request to the URL `https://52.26.65.149:8080/passenger/5`. The response status is `200 OK` with a time of `181 ms`. The response body is displayed in XML format:

```
1 <SuccessfulResponse>
2   <Response>
3     <code>200</code>
4     <message>Passenger with id 5 is deleted successfully</message>
5   </Response>
6 </SuccessfulResponse>
```

Delete passenger (invalid passenger id). Example for URL is below

<https://52.26.65.149:8080/passenger/99>



## Flight Services

Fetch flight using flight number. Example for URL is below

<https://52.26.65.149:8080/flight/AA2>

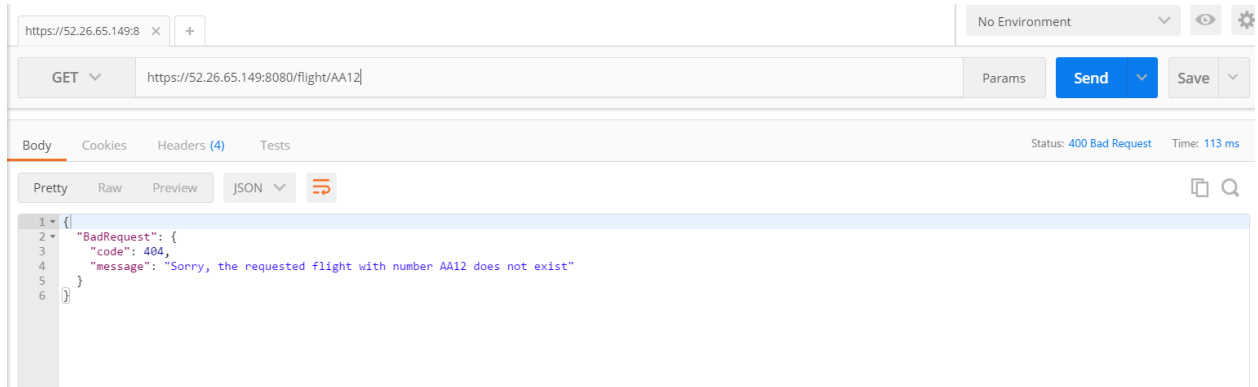
The screenshot shows a web browser with a REST client interface. The address bar shows the URL `https://52.26.65.149:8080/flight/AA2`. The method is `GET`. The response status is `200 OK` and the time taken is `168 ms`. The response body is a JSON object, displayed in the "Pretty" view. The JSON object contains the following data:

```
{
  "flight": {
    "price": "130",
    "from": "Hyd",
    "to": "Del",
    "departureTime": "2017-04-27-06",
    "arrivalTime": "2017-04-27-09",
    "seatsLeft": "99",
    "description": "HyderabadtoDelhi",
    "plane": {
      "capacity": "100",
      "model": "350",
      "manufacturer": "Airbus",
      "yearOfManufacture": "1995"
    },
    "passengers": [
      {
        "id": "3",
        "firstname": "Prudhvi",
        "lastname": "Raj",
        "age": "26",
        "gender": "male",
        "phone": "992123"
      }
    ]
  },
  "number": "AA2"
}
```



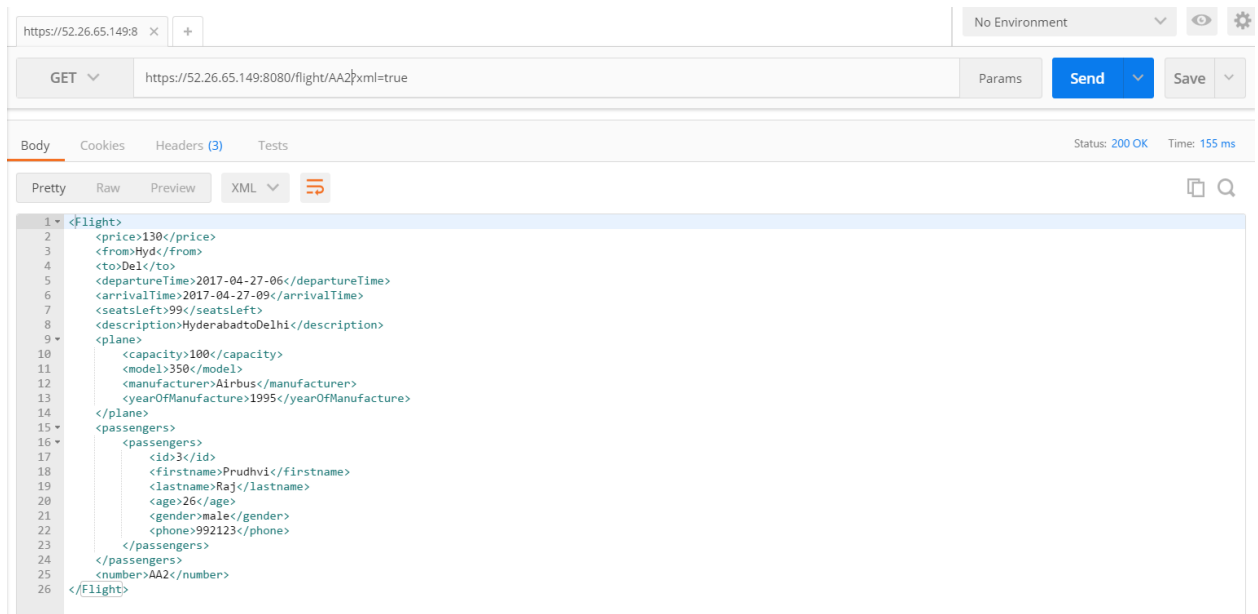
Fetch flight using invalid or nonexistent flight number. Example for URL is below

<https://52.26.65.149:8080/flight/AA12>



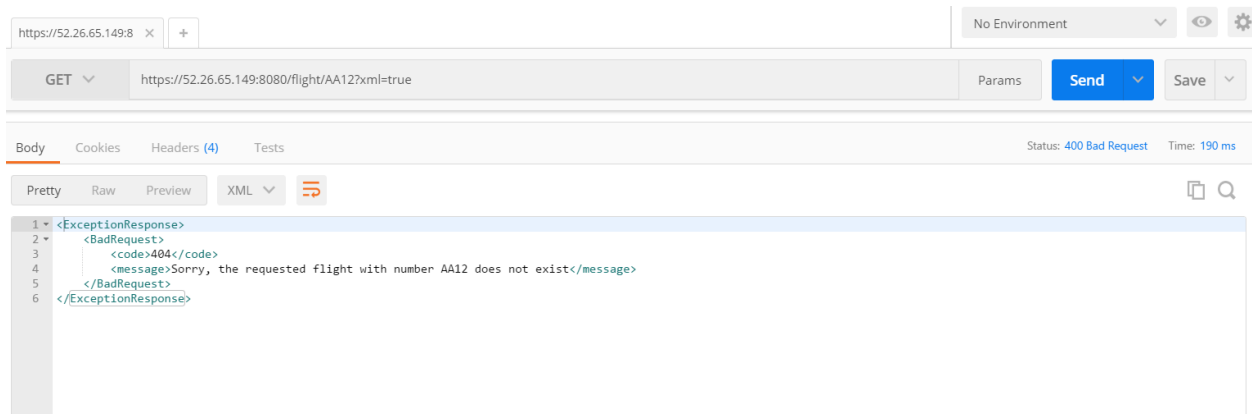
Fetch flight for xml. Example for URL is below

<https://52.26.65.149:8080/flight/AA2?xml=true>



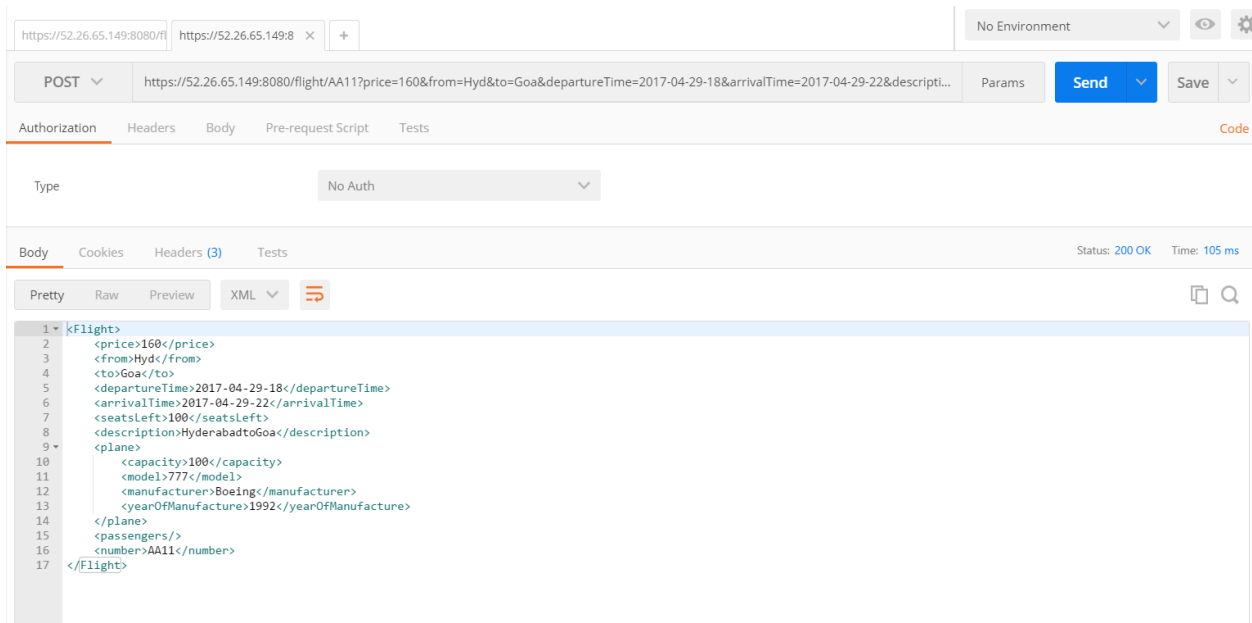
Fetch flight for xml invalid flight number. Example for URL is below

<https://52.26.65.149:8080/flight/AA12?xml=true>



Create flight. Example for URL is below

<https://52.26.65.149:8080/flight/AA11?price=160&from=Hyd&to=Goa&departureTime=2017-04-29-18&arrivalTime=2017-04-29-22&description=HyderabadtoGoa&capacity=100&model=777&manufacturer=Boeing&yearOfManufacture=1992>



Update flight using flight number. Example for URL is below

https://52.26.65.149:8080/flight/AA12?price=160&from=Hyd&to=Goa&departureTime=2017-04-29-18&arrivalTime=2017-04-29-22&description=HyderabadtoGoa&capacity=100&model=777&manufacturer=Boeing&yearOfManufacture=1992

The screenshot shows a REST client interface with a POST request to `https://52.26.65.149:8080/flight/AA12?price=160&from=Hyd&to=Goa&departureTime=2017-04-29-18&arrivalTime=2017-04-29-22&description=HyderabadtoGoa&capacity=100&model=777&manufacturer=Boeing&yearOfManufacture=1992`. The response is a 200 OK status with a time of 172 ms. The response body is displayed in XML format:

```
1 <Flight>
2   <price>160</price>
3   <from>Hyd</from>
4   <to>Goa</to>
5   <departureTime>2017-04-29-19</departureTime>
6   <arrivalTime>2017-04-29-22</arrivalTime>
7   <seatsLeft>100</seatsLeft>
8   <description>HyderabadtoGoa</description>
9   <plane>
10    <capacity>100</capacity>
11    <model>777</model>
12    <manufacturer>Boeing</manufacturer>
13    <yearOfManufacture>1992</yearOfManufacture>
14  </plane>
15  <passengers/>
16  <number>AA12</number>
17 </Flight>
```

Update flight with invalid data( capacity less than reservation count). Example for URL is below

https://52.26.65.149:8080/flight/AA14?price=160&from=Hyd&to=Goa&departureTime=2017-04-30-14&arrivalTime=2017-04-30-19&description=HyderabadtoGoa&capacity=3&model=777&manufacturer=Boeing&yearOfManufacture=1992

The screenshot shows a REST client interface with a POST request to `https://52.26.65.149:8080/flight/AA14?price=160&from=Hyd&to=Goa&departureTime=2017-04-30-14&arrivalTime=2017-04-30-19&description=HyderabadtoGoa&capacity=3&model=777&manufacturer=Boeing&yearOfManufacture=1992`. The response is a 400 Bad Request status with a time of 180 ms. The response body is displayed in XML format:

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>400</code>
4     <message>Reservation count will be higher than changed capacity.</message>
5   </BadRequest>
6 </ExceptionResponse>
```

Delete flight using flight number. Example for URL is below

<https://52.26.65.149:8080/flight/AA12>

The screenshot shows a REST client interface with the following details:

- URL Bar:** Contains the URL `https://52.26.65.149:8080/flight/AA12`.
- Method:** Set to `DELETE`.
- Environment:** Set to `No Environment`.
- Authorization:** Set to `No Auth`.
- Status:** `200 OK` (Status: 200 OK, Time: 163 ms).
- Body:** The response is displayed in XML format:

```
1 <SuccessfulResponse>
2   <Response>
3     <code>200</code>
4     <message>Flight with number AA12 is deleted successfully</message>
5   </Response>
6 </SuccessfulResponse>
```

Delete flight using invalid or nonexistent flight number. Example for URL is below

<https://52.26.65.149:8080/flight/AA17>

The screenshot shows a REST client interface with the following details:

- URL Bar:** Contains the URL `https://52.26.65.149:8080/flight/AA17`.
- Method:** Set to `DELETE`.
- Environment:** Set to `No Environment`.
- Authorization:** Set to `No Auth`.
- Status:** `400 Bad Request` (Status: 400 Bad Request, Time: 111 ms).
- Body:** The response is displayed in XML format:

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>404</code>
4     <message>Flight with number AA17 does not exist</message>
5   </BadRequest>
6 </ExceptionResponse>
```

Delete flight for existing reservations. Example for URL is below

https://52.26.65.149:8080/flight/AA2

The screenshot shows a REST client interface with the following details:

- URL Bar:** Contains two tabs. The first tab is active and shows the URL `https://52.26.65.149:8080/flight/AA2`. The second tab is inactive and shows `https://52.26.65.149:8`.
- Method:** A dropdown menu is set to `DELETE`.
- Params:** A button labeled `Send` is visible next to the `Params` tab.
- Authorization:** A dropdown menu is set to `No Auth`.
- Body:** The `Body` tab is selected. It shows the response in `XML` format. The response is an `ExceptionResponse` containing a `BadRequest` with a `400` status code and a message: `There are existing reservations for this flight.`
- Status Bar:** At the bottom right, it indicates `Status: 400 Bad Request` and `Time: 103`.

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>400</code>
4     <message>There are existing reservations for this flight.</message>
5   </BadRequest>
6 </ExceptionResponse>
```

## Reservation Services

Fetch reservation using reservation id. Example for URL is below

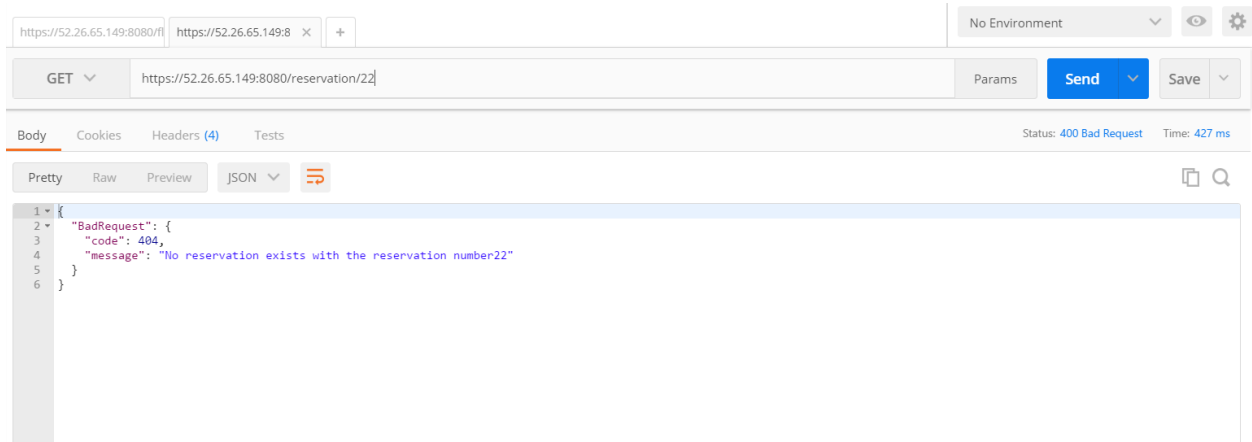
<https://52.26.65.149:8080/reservation/2>

The screenshot shows a web browser window with a REST client interface. The address bar displays the URL `https://52.26.65.149:8080/reservation/2`. The method is set to `GET`. The response status is `200 OK` and the time taken is `188 ms`. The response body is displayed in JSON format, showing the details of a reservation with ID 2.

```
1 {
2   "orderId": "2",
3   "price": 250,
4   "passenger": {
5     "id": "3",
6     "firstname": "Prudhvi",
7     "lastname": "Raj",
8     "age": "26",
9     "gender": "male",
10    "phone": "992123"
11  },
12  "flights": [
13    {
14      "price": "120",
15      "from": "Hyd",
16      "to": "Ban",
17      "departureTime": "2017-04-27-21",
18      "arrivalTime": "2017-04-27-23",
19      "seatsLeft": "99",
20      "description": "HyderabadtoBanglore",
21      "plane": {
22        "capacity": "100",
23        "model": "777",
24        "manufacturer": "Boeing",
25        "yearOfManufacture": "1997"
26      },
27      "number": "AA1"
28    },
29    {
30      "price": "130",
31      "from": "Hyd",
32      "to": "Del",
33      "departureTime": "2017-04-27-06",
34      "arrivalTime": "2017-04-27-09",
35      "seatsLeft": "99",
36      "description": "HyderabadtoDelhi",
37      "plane": {
38        "capacity": "100",
39        "model": "777",
40        "manufacturer": "Boeing",
41        "yearOfManufacture": "1997"
42      },
43      "number": "AA2"
44    }
45  ]
46 }
```

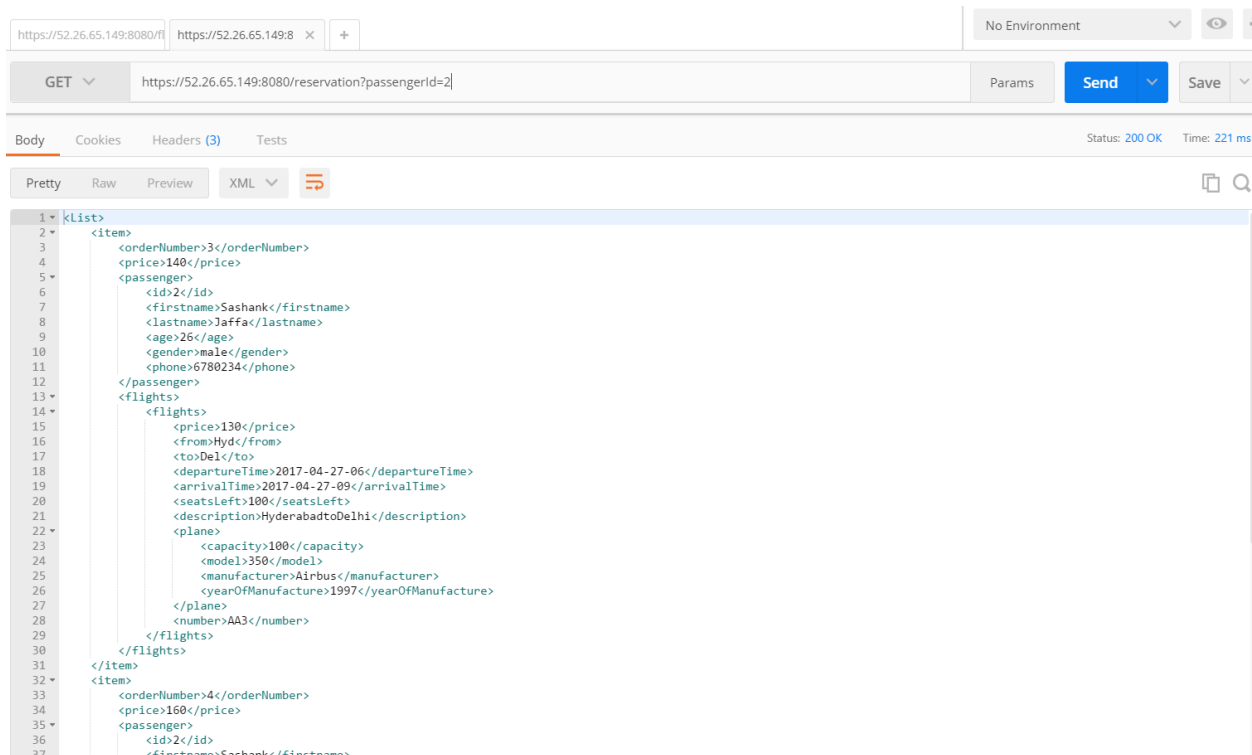
Fetch reservation using order number (non-existent order number ). Example for URL is below

<https://52.26.65.149:8080/reservation/22>



Search reservation using passenger id. Example for URL is below

<https://52.26.65.149:8080/reservation?passengerId=2>



Search reservation using from and to. Example for URL is below

<https://52.26.65.149:8080/reservation?from=Hyd&to=Ban>

```
Body Cookies Headers (3) Tests Status: 200 OK Time:
Pretty Raw Preview XML
1 <list>
2   <item>
3     <orderNumber>6</orderNumber>
4     <price>240</price>
5     <passenger>
6       <id>6</id>
7       <firstname>David</firstname>
8       <lastname>Trump</lastname>
9       <age>11</age>
10      <gender>male</gender>
11      <phone>1234567</phone>
12    </passenger>
13    <flights>
14      <flights>
15        <price>120</price>
16        <from>Hyd</from>
17        <to>Ban</to>
18        <departureTime>2017-04-28-13</departureTime>
19        <arrivalTime>2017-04-28-15</arrivalTime>
20        <seatsLeft>99</seatsLeft>
21        <description>HyderabadtoBangalore</description>
22        <plane>
23          <capacity>100</capacity>
24          <model>777</model>
25          <manufacturer>Boeing</manufacturer>
26          <yearOfManufacture>1997</yearOfManufacture>
27        </plane>
28        <number>AA8</number>
29      </flights>
30    </flights>
31  </item>
32 </list>
```

Search reservation using flight number. Example for URL is below

<https://52.26.65.149:8080/reservation?flightNumber=AA8>

```
Body Cookies Headers (3) Tests Status: 200 OK Time: 159 ms
Pretty Raw Preview XML
1 <list>
2   <item>
3     <orderNumber>2</orderNumber>
4     <price>250</price>
5     <passenger>
6       <id>3</id>
7       <firstname>Prudhvi</firstname>
8       <lastname>Raj</lastname>
9       <age>26</age>
10      <gender>male</gender>
11      <phone>992123</phone>
12    </passenger>
13    <flights>
14      <flights>
15        <price>120</price>
16        <from>Hyd</from>
17        <to>Ban</to>
18        <departureTime>2017-04-27-21</departureTime>
19        <arrivalTime>2017-04-27-23</arrivalTime>
20        <seatsLeft>99</seatsLeft>
21        <description>HyderabadtoBangalore</description>
22        <plane>
23          <capacity>100</capacity>
24          <model>777</model>
25          <manufacturer>Boeing</manufacturer>
26          <yearOfManufacture>1997</yearOfManufacture>
27        </plane>
28        <number>AA1</number>
29      </flights>
30      <flights>
31        <price>130</price>
32        <from>Hyd</from>
33        <to>Del</to>
34        <departureTime>2017-04-27-06</departureTime>
35        <arrivalTime>2017-04-27-09</arrivalTime>
36        <seatsLeft>99</seatsLeft>
37        <description>HyderabadtoDelhi</description>
```



Create new reservation. Example for URL is below

<https://52.26.65.149:8080/reservation?passengerId=6&flightLists=AA5,AA8>

The screenshot shows a REST client interface with the following details:

- URL:** `https://52.26.65.149:8080/reservation?passengerId=6&flightLists=AA5,AA8`
- Method:** POST
- Status:** 200 OK
- Time:** 189 ms
- Body (XML):**

```
1 <Reservation>
2   <orderNumber>6</orderNumber>
3   <price>280</price>
4   <passenger>
5     <id>6</id>
6     <firstname>David</firstname>
7     <lastname>Trump</lastname>
8     <age>11</age>
9     <gender>male</gender>
10    <phone>1234567</phone>
11  </passenger>
12  <flights>
13    <flight>
14      <price>160</price>
15      <from>Hyd</from>
16      <to>Goa</to>
17      <departureTime>2017-04-27-18</departureTime>
18      <arrivalTime>2017-04-27-22</arrivalTime>
19      <seatsLeft>97</seatsLeft>
20      <description>HyderabadtoGoa</description>
21      <plane>
22        <capacity>100</capacity>
23        <model>777</model>
24        <manufacturer>Boeing</manufacturer>
25        <yearOfManufacture>1992</yearOfManufacture>
26      </plane>
27      <number>AA5</number>
28    </flight>
29  </flights>
30  <flights>
31    <price>120</price>
32    <from>Hyd</from>
33    <to>Ban</to>
34    <departureTime>2017-04-28-13</departureTime>
35    <arrivalTime>2017-04-28-15</arrivalTime>
36    <seatsLeft>99</seatsLeft>
37    <description>HyderabadtoBangalore</description>
38    <plane>
```

Create new reservation for overlapped flights (Throw error). Example for URL is below

<https://52.26.65.149:8080/reservation?passengerId=7&flightLists=AA5,AA4>

The screenshot shows a REST client interface with the following details:

- URL:** `https://52.26.65.149:8080/reservation?passengerId=7&flightLists=AA5,AA4`
- Method:** POST
- Status:** 400 Bad Request
- Time:** 161 ms
- Body (XML):**

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>404</code>
4     <message>Flights AA5 AA4 overlap in timings</message>
5   </BadRequest>
6 </ExceptionResponse>
```

Update Reservation on particular order number. Example for URL is below

<https://52.26.65.149:8080/reservation/6&flightsAdded=AA7,AA8&flightsRemoved=AA5>

The screenshot shows a REST client interface with the following details:

- URL:** `https://52.26.65.149:8080/reservation/6?flightsAdded=AA7,AA8&flightsRemoved=AA5`
- Method:** POST
- Status:** 200 OK
- Time:** 199 ms
- Body:** A JSON object representing a reservation update:

```
1 {
2   "orderNumber": "6",
3   "price": 240,
4   "passenger": {
5     "id": "6",
6     "firstname": "David",
7     "lastname": "Trump",
8     "age": "11",
9     "gender": "male",
10    "phone": "1234567"
11  },
12  "flights": [
13    {
14      "price": "120",
15      "from": "Hyd",
16      "to": "Ban",
17      "departureTime": "2017-04-28-13",
18      "arrivalTime": "2017-04-28-15",
19      "seatsLeft": "99",
20      "description": "HyderabadtoBanglore",
21      "plane": {
22        "capacity": "100",
23        "model": "777",
24        "manufacturer": "Boeing",
25        "yearOfManufacture": "1997"
26      },
27      "number": "AA8"
28    },
29    {
30      "price": "120",
31      "from": "Hyd",
32      "to": "Ban",
33      "departureTime": "2017-04-28-08",
34      "arrivalTime": "2017-04-28-11",
35      "seatsLeft": "99",
36      "description": "HyderabadtoBanglore",
37      "plane": {
```

Update Reservation on invalid order number(throw error). Example for URL is below

<https://52.26.65.149:8080/reservation/6?flightsAdded=AA12,AA13&flightsRemoved=AA7>

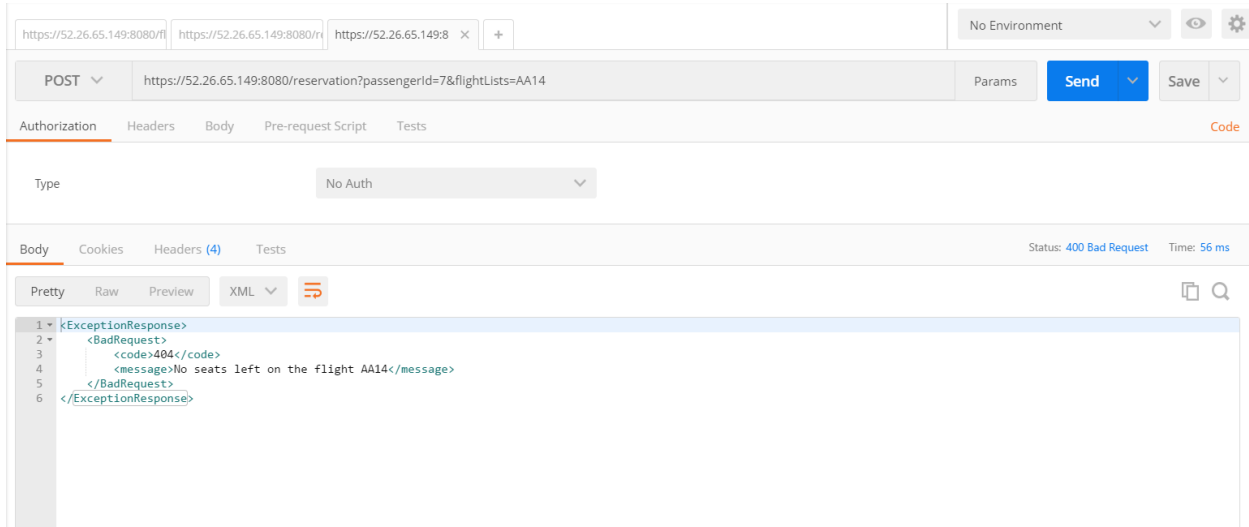
The screenshot shows a REST client interface with the following details:

- URL:** `https://52.26.65.149:8080/reservation/6?flightsAdded=AA12,AA13&flightsRemoved=AA7`
- Method:** POST
- Status:** 400 Bad Request
- Time:** 161 ms
- Body:** A JSON object representing a 404 error:

```
1 {
2   "BadRequest": {
3     "code": 404,
4     "message": "Flight AA12 does not exist"
5   }
6 }
```

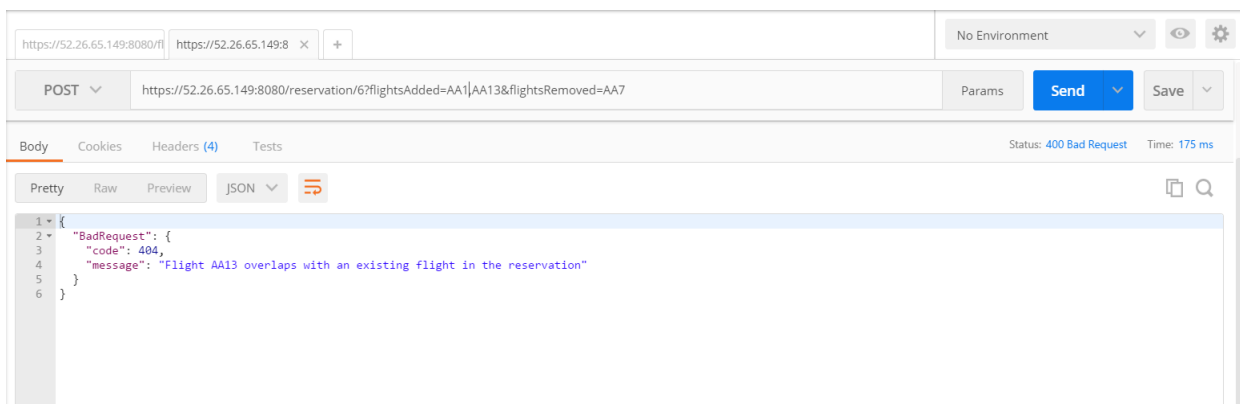
Update Reservation exceeding flight seats (throws error). Example for URL is below

<https://52.26.65.149:8080/reservation?passengerId=7&flightLists=AA14>



Update Reservation on particular order number for overlapping flights (throw error). Example for URL is below

<https://52.26.65.149:8080/reservation/6?flightsAdded=AA1,AA13&flightsRemoved=AA7>



Delete reservation particular order number. Example for URL is below

<https://52.26.65.149:8080/reservation/6>

The screenshot shows a REST client interface with the following details:

- URL Bar:** Contains the URL `https://52.26.65.149:8080/reservation/6`.
- Method:** Set to `DELETE`.
- Status:** `200 OK` with a response time of `185 ms`.
- Response Body (XML):**

```
1 <SuccessfulResponse>
2   <Response>
3     <code>200</code>
4     <message>Reservation with number 6 is canceled successfully is deleted successfully</message>
5   </Response>
6 </SuccessfulResponse>
```

Delete reservation invalid order number(throws error). Example for URL is below

<https://52.26.65.149:8080/reservation/10>

The screenshot shows a REST client interface with the following details:

- URL Bar:** Contains the URL `https://52.26.65.149:8080/reservation/10`.
- Method:** Set to `DELETE`.
- Status:** `400 Bad Request` with a response time of `114 ms`.
- Response Body (XML):**

```
1 <ExceptionResponse>
2   <BadRequest>
3     <code>404</code>
4     <message>Reservation with order number 10 does not exist.</message>
5   </BadRequest>
6 </ExceptionResponse>
```