BAB IV
IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas mengenai implementasi Domain Driven Design pada pengembangan REST Web service dalam studi kasus reservasi tiket pesawat terbang yang diujicobakan untuk maskapai Sriwijaya Air.

4.1 Batasan Implementasi

Batasan implementasi yang dilakukan dalam pengembangan REST Webservice hanya mengimplementasikan 4 fitur ke dalam reservasi tiket pesawat yaitu adalah:

1. **Search**

   Merupakan fitur yang ada dalam reservasi tiket pesawat yang fungsinya untuk melakukan pencarian penerbangan.

2. **Book**

   Merupakan fitur yang ada dalam reservasi tiket pesawat yang digunakan setelah proses search selesai, book sendiri merupakan proses pemesanan dan pengisian detail penumpang, dengan mengisikan *contact person* dan nama-nama penumpang yang telah dikategorikan menjadi (dewasa, anak-anak, dan bayi).

3. **Payment**

   Merupakan fitur yang ada dalam reservasi tiket pesawat terbang yang berfungsi untuk melakukan proses pembayaran dan dikhususkan untuk pembayaran melalui ATM.

4. **Book Info**

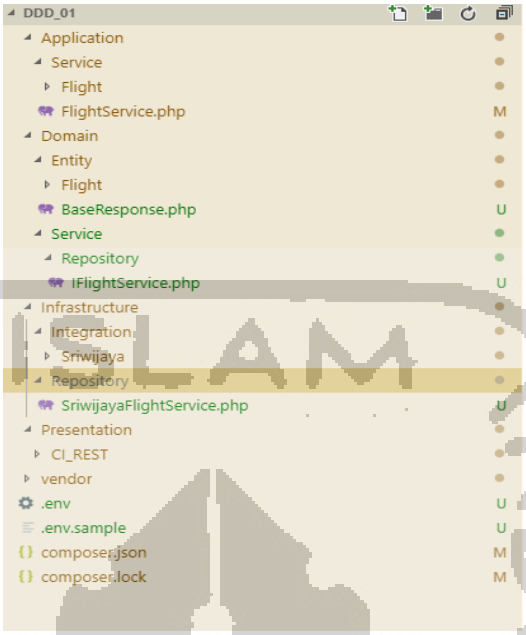
   Merupakan fitur yang ada dalam reservasi tiket pesawat yang berfungsi untuk mendapatkan info kode booking setelah melalui proses-proses yang ada diatas.

4.2 Implementasi

Berikut merupakan hasil implementasi Domain Driven Design pada pengembangan REST Web Servive dengan studi kasus reservasi tiket pesawat terbang.

4.2.1. Modules REST Web Service

Module yang telah dibuat dengan implementasi Domain Driven Design adalah dengan mengategorikan menjadi 4 level, pada level Presentation Layer terdapat module Presentation yang berisi CI_REST untuk merepresentasikan informasi kepada pengguna, pada level Application Layer terdapat module Application yang berisi Flight Service berfungsi untuk mengkoordinasikan aktifitas aplikasi, kemudian pada level Domain Layer berisi *entity* yang mengandung informasi tentang *domain* dan merupakan pusat bisnis aplikasi. Sedangkan pada level Infrastructure Layer terdapat modul Infrastructure yang berisi *repository* dan *integration* berfungsi sebagai *supporting library*. Selengkapnya dapat dilihat pada Gambar 4.1.



Gambar 4.1 Modules Implementasi DDD REST Webservice

4.2.2. Implementasi Fitur Search

Tahap ini adalah implementasi dari fitur search yang berisi *Search Request* yang digunakan untuk permintaan pencarian jadwal penerbangan, dan *Search Response* berfungsi untuk merespon permintaan berupa data kembalian yang menampilkan detail jadwal penerbangan. Selengkapnya dapat dilihat pada Tabel 4.1.

Tabel 4.1 Search Request dan Search Response

Endpoint	`http://hostname/api/v1.0/flight/search`			
HTTP Method	POST			
Request				
	#	**Property**	**Type**	**Description**
	1	Request Object	Json	Proses permintaan pencarian penerbangan

	<pre>{
	"airportFrom": "JOG",
	"airportTo": "BTH",
	"departDate": "2019-09-20",
	"returnStatus": true,
	"returnDate": "2019-09-25",</pre>			

<table>
<tr><td></td><td>

```
  "promoCode": "",
  "paxCount": {
    "adult": 5,
    "child": 4,
    "infant": 3
  }
}
```

</td></tr>
<tr><td colspan="2">Response</td></tr>
<tr><td></td><td>

#	Property	Type	Description
1	Response Object	Json	Proses respon permintaan dengan kembalian data berupa jadwal penerbangan

</td></tr>
<tr><td></td><td>

```
{
    "data": {
        "searchKey":
"020800138779688080837965888778728684897670656872",
        "stiStatus": "YES",
        "timeInt": 900,
        "availableTrips": [
            {
                "tripType": "Departure",
                "airportFrom": "JOG",
                "airportTo": "BTH",
                "journeys": [
                    {
                        "airportFrom": "JOG",
                        "cityFromName": "Yogyakarta",
                        "airportTo": "BTH",
                        "cityToName": "Batam",
                        "std": "2019-09-20 05.00.00 AM",
                        "stdShort": "2019-09-20 05.00",
                        "sta": "2019-09-20 08.35.00 AM",
                        "staShort": "2019-09-20 08.35",
                        "flightTime": 215,
                        "description": "",
                        "journeyKey": "SJ-219--JOG-CGK--2019-09-
20--05-00~SJ-032--CGK-BTH--2019-09-20--07-00",
                        "segments": [….],
                        "availableCabinClasses": […]
                    },
                    {….}
                ]
            },
            {
                "tripType": "Return",
                "airportFrom": "BTH",
                "airportTo": "JOG",
                "journeys": [
                    {
                        "airportFrom": "BTH",
                        "cityFromName": "Batam",
                        "airportTo": "JOG",
                        "cityToName": "Yogyakarta",
                        "std": "2019-09-25 01.05.00 PM",
                        "stdShort": "2019-09-25 13.05",
                        "sta": "2019-09-25 06.55.00 PM",
                        "staShort": "2019-09-25 18.55",
                        "flightTime": 350,
                        "description": "",
                        "journeyKey": "SJ-033--BTH-CGK--2019-09-
25--13-05~SJ-234--CGK-JOG--2019-09-25--17-45",
                        "segments":[…..],
                        "availableCabinClasses": […]
                    },
                    {….}
                ]
```

</td></tr>
</table>

<table>
<tr><td></td><td></td><td><pre>
            }
        ]
    },
    "errorCode": "EC:0000",
    "errorMessage": "Success."
}
</pre></td></tr>
</table>

4.2.3. Implementasi Fitur Book

Tahap ini adalah implementasi dari fitur *book* yang berisi *Book Request* yang digunakan untuk permintaan pengisian detail penumpang, data yang harus dimasukan adalah *journey* (mengisikan informasi penerbangan), *paxCount* (mengisikan jumlah penumpang), *contact* (mengisikan informasi kontak), *paxs* (mengisikan detail nama-nama penumpang), dan *itinerarries* (mengisikan *searchkey* yang diperoleh dari Search Response). *Book Response* berfungsi untuk merespon permintaan berupa data kembalian yang menampilkan detail pemesanan berupa detail penumpang dan detail perjalanan penerbangan. Selengkapnya dapat dilihat pada Tabel 4.2 berikut ini.

Tabel 4.2 Book Request dan Book Response

<table>
<tr><td>Endpoint</td><td colspan="4">http://hostname/api/v1.0/flight/book</td></tr>
<tr><td>HTTP Method</td><td colspan="4">POST</td></tr>
<tr><td>Request</td><td colspan="4"></td></tr>
<tr><td></td><td>#</td><td>Property</td><td>Type</td><td>Description</td></tr>
<tr><td></td><td>1</td><td>Request Object</td><td>Json</td><td>Proses permintaan pengisian detail penumpang</td></tr>
<tr><td></td><td colspan="4"><pre>
{
  "journey": {
    "airportFrom": "JOG",
    "airportTo": "BTH",
    "departDate": "2019-09-20",
    "returnStatus": true,
    "returnDate": "2019-09-25",
    "promoCode": "",
    "paxCount": {
      "adult": 5,
      "child": 4,
      "infant": 3
    }
  },
  "contact": {
    "contactName": "AGUNGBUDI",
    "contactEmail": "AGUNG.UI09@GMAIL.COM",
    "contactPhone": "090897779"
  },
  "paxs": {
    "adults": [
</pre></td></tr>
</table>

```
        {
          "suffix": "MR",
          "firstName": "AGUNG",
          "lastName": "BUDI",
          "FFNumber": "233"
        },{
          "suffix": "MRS",
          "firstName": "Jingga",
          "lastName": "Ratna",
          "FFNumber": "454"
        },{
          "suffix": "MR",
          "firstName": "Rudi",
          "lastName": "Santosa",
          "FFNumber": "232"
        },{
          "suffix": "MR",
          "firstName": "Ibrahim",
          "lastName": "Samad",
          "FFNumber": "235"
        },{
          "suffix": "MRS",
          "firstName": "Via",
          "lastName": "Rahma",
          "FFNumber": "6565"
        }
      ],
      "childs": [
        {
          "suffix": "MSTR",
          "firstName": "Yudha",
          "lastName": "Mubarok",
          "FFNumber": "12",
          "dateOfBirth": "2012-05-27"
        },
        {
          "suffix": "MISS",
          "firstName": "Armin",
          "lastName": "Vahira",
          "FFNumber": "98",
          "dateOfBirth": "2013-05-27"
        },
        {
          "suffix": "MISS",
          "firstName": "Citra",
          "lastName": "Putri",
          "FFNumber": "87",
          "dateOfBirth": "2010-05-27"
        },
        {
          "suffix": "MISS",
          "firstName": "Indah",
          "lastName": "Permata",
          "FFNumber": "2",
          "dateOfBirth": "2012-05-27"
        }

      ],
      "infants": [
        {
          "suffix": "MISS",
          "firstName": "Malika",
          "lastName": "Putri",
          "dateOfBirth": "2018-12-08",
          "adultAssoc": 1
        },
        {
          "suffix": "MISS",
          "firstName": "Yulaikha",
          "lastName": "Putri",
```

```
        "dateOfBirth": "2019-01-17",
        "adultAssoc": 2
      },
      {
        "suffix": "MISS",
        "firstName": "Aisha",
        "lastName": "Putri",
        "dateOfBirth": "2018-11-18",
        "adultAssoc": 3
      }
    ]
  },
  "selectedItineraries": {
    "key": "02080013877968808083796588877872868489767065687 2",
    "journeys": [
      {
        "category": "Departure",
        "key": "SJ-219--JOG-CGK--2019-09-20--05-00~SJ-032--CGK-BTH--
2019-09-20--07-00",
        "selectedCabin": [
          {
            "subClass": "V",
            "key": "25231825:V:S"
          },
          {
            "subClass": "Y",
            "key": "19493363:Y:S"
          }
        ]
      },
      {
        "category": "Return",
        "key": "SJ-033--BTH-CGK--2019-09-25--13-05~SJ-234--CGK-JOG--
2019-09-25--17-45",
        "selectedCabin": [
          {
            "subClass": "Y",
            "key": "19509555:Y:S"
          },
          {
            "subClass": "V",
            "key": "20275056:V:S"
          }
        ]
      }
    ]
  }
}
```


Response

	#	Property	Type	Description
	1	Response Object	Json	Merespon permintaan dengan menampilkan data penerbangan dan detail penumpang


```
{
    "data": {
        "bookingCode": "BKLJRJ",
        "numericBookingCode": null,
        "promoCode": null,
        "paymentMethod": null,
        "statusShow": null,
        "checkInFlag": null,
        "checkInDate": null,
        "itineraryDetails": {
            "reservationDetails": {
                "bookingCode": "BKLJRJ",
                "bookingDate": "07 Aug 2019 17:39 (GMT+7)",
```

```
                    "currencyCode": "IDR",
                    "balanceDue": 39945600,
                    "balanceDueRemarks": "*Extra Cover Insurance
(STI) not include in balance due.",
                    "bookingStatus": "Hold",
                    "timeInfo": "07 Aug 2019 18:39 (GMT+7)",
                    "timeInfoDescription": "TimeLimit"
                },
                "passengerDetails":[…]
                "itineraries":[…]
                    "paymentDetails": {
                    "currencyCode": "IDR",
                    "basicFare": 35196000,
                    "otherFare": 4749600,
                    "STIFare": 0,
                    "totalFare": 39945600,
                    "directVoucher": 0,
                    "addOnFare": 0,
                    "ntaFare": 39264480
                },
                "contactList": [
                    {
                        "type": "Phone",
                        "description": "Main",
                        "value": "090897779"
                    },
                    {
                        "type": "Email",
                        "description": "Work",
                        "value": "AGUNG.UI09@GMAIL.COM"
                    }
                ],
                "agentDetails": {
                    "bookedBy": "USERNAME",
                    "issuedBy": "-"
                },
                "bookingRemarks": [],
                "additionalInformation": {}
            }
        },
        "errorCode": "EC:0000",
        "errorMessage": "Success."
}
```


4.2.4. Implementasi Fitur Payment

Tahap ini adalah implementasi dari fitur *Payment* yang berisi *Payment Request* yang digunakan untuk permintaan pengisian metode pembayaran lewat ATM, dan *Payment Response* berfungsi untuk merespon permintaan berupa data kembalian yang menampilkan kode pembayaran. Selengkapnya dapat dilihat pada Tabel 4.3 berikut ini.

Tabel 4.3 Payment Request dan Payment Response

Endpoint	`http://hostname/api/v1.0/flight/set_payment`			
HTTP Method	POST			
Request				
	#	**Property**	**Type**	**Description**

	1	Request Object	Json	Proses permintaan pengisian metode pembayaran lewat ATM


```
{
  "bookingCode": "BKLJRJ",
  "payment": {
    "paymentMethod": "ATM-BCA",
    "paymentCardName": "Agung Doe",
    "paymentTenor": "",
    "paymentEmail": "customer.sriwijaya@email.com",
    "paymentContact": {
      "email": "customer.sriwijaya@email.com",
      "phone": "6208123321123"
    }
  },
  "FFInfo": {
    "FFType": "",
    "FFNumber": "",
    "FFPassword": "",
    "FFPoint": ""
  }
}
```


Response

	#	Property	Type	Description
	1	Response Object	Json	Merespon permintaan dengan kembalian data berupa kode pembayaran


```
{
    "data": [
        {
            "paymentCode": "7779037296452",
            "paymentMethod": "ATM-BCA",
            "paymentMethodDescription": "ATM",
            "paymentAmount": 39945600,
            "FFDetail": null,
            "specialRequest": "",
            "BNIWOW": "NO"
        }
    ],
    "errorCode": "EC:0000",
    "errorMessage": "Success"
}
```


4.2.5. Implementasi Fitur Booking Info

Tahap ini adalah implementasi dari fitur *Booking Info* yang berisi *Booking Info Request* yang digunakan untuk permintaan pengisian kode booking dan *Payment Response* berfungsi untuk merespon permintaan berupa data kembalian yang menampilkan detail booking info. Selengkapnya dapat dilihat pada Tabel 4.4 berikut ini.

Tabel 4.4 Booking Info Request dan Booking Info Response

Endpoint	`http://hostname/api/v1.0/flight/get_book_info`			
HTTP Method	POST			
Request				
	#	**Property**	**Type**	**Description**
	1	Request Object	Json	Proses permintaan pengisian kode booking

	<pre>{
	"bookingCode": "BKLJRJ"
	}</pre>			

Response				
	#	**Property**	**Type**	**Description**
	1	Response Object	Json	Merespon permintaan dengan kembalian data berupa detail booking info

	<pre>{
	"data": {
	"bookingCode": "BKLJRJ",
	"numericBookingCode": "7779037296452",
	"promoCode": null,
	"paymentMethod": "ATM",
	"statusShow": "Need Payment",
	"checkInFlag": "",
	"checkInDate": "",
	"itineraryDetails": {
	"reservationDetails": {
	"bookingCode": "BKLJRJ",
	"bookingDate": "07 Aug 2019 17:39 (GMT+7)",
	"currencyCode": "IDR",
	"balanceDue": 39945600,
	"balanceDueRemarks": "*Extra Cover Insurance (STI) not include in balance due.",
	"bookingStatus": "Hold",
	"timeInfo": "07 Aug 2019 18:39 (GMT+7)",
	"timeInfoDescription": "TimeLimit"
	},
	"passengerDetails": [….]
	"itineraries": {
	"journeys": [
	{
	"category": "Departure",
	"segments":[….]
	},
	{
	"category": "Return",
	"segments":[….]
	}
	]
	},
	"paymentDetails": {
	"currencyCode": "IDR",
	"basicFare": 35196000,
	"otherFare": 4749600,</pre>			

```
                    "STIFare": 0,
                    "totalFare": 39945600,
                    "directVoucher": 0,
                    "addOnFare": 0,
                    "ntaFare": 39264480
                },
                "contactList": [
                    {
                        "type": "Phone",
                        "description": "Main",
                        "value": "090897779"
                    },
                    {
                        "type": "Email",
                        "description": "Work",
                        "value": "AGUNG.UI09@GMAIL.COM"
                    }
                ],
                "agentDetails": {
                    "bookedBy": "USERNAME",
                    "issuedBy": "-"
                },
                "bookingRemarks": [],
                "additionalInformation": {}
            }
        },
        "errorCode": "EC:0000",
        "errorMessage": "Success."
    }
```


4.3 Pengujian

Setelah melewati tahap analisis, perancangan dan implementasi maka suatu sistem perangkat lunak haruslah melewati tahap *testing* (pengujian) sebelum aplikasi perangkat lunak tersebut digunakan *client.* Pengujian akan dilakukan dengan menggunakan metode pengujian *black box*, pengujian *black box* ini menitikberatkan pada fungsi sistem. Hal ini dimaksudkan agar aplikasi perangkat lunak tersebut dapat berjalan sesuai dengan keinginan dan untuk meminimalisir terjadinya error. Hal yang cukup penting sebelum aplikasi perangkat lunak digunakan *client* adalah adanya fasilitas *error reporting* untuk mengirim *error/bug* yang terjadi kepada *developer* untuk segera mendapatkan *action*, sehingga akan menjadi aplikasi yang *free bug* .

4.3.1 Pengujian Search

Pengujian terhadap fitur Search dilakukan pada sisi *developer* untuk menguji fungsi dan kinerja fitur. Pengujian fitur tersebut dilakukan dengan memberikan *input* pada *Search Request* sesuai dengan perancangan dan implementasi yang telah

dibuat. Hasil pengujian dilihat berdasarkan hasil keluaran dari *Search Response* setelah mendapatkan dua jenis pengujian yaitu pengujian normal dan pengujian tidak normal. Adapun hasil pengujian fitur Search dapat dilihat pada Tabel 4.5.

Tabel 4.5 Pengujian Fitur Search

Pengujian Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**

1	Memberi masukan dengan benar:
	{
	"airportFrom": "JOG",
	"airportTo": "BTH",
	"departDate": "2019-09-20",
	"returnStatus": true,
	"returnDate": "2019-09-25",
	"promoCode": "",
	"paxCount": {
	"adult": 5,
	"child": 4,
	"infant": 3
	}
	}
		Menampilkan Hasil Search Response	Hasil Search Response ditampilkan	Valid

Pengujian Tidak Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Mengosongkan returnDate: “ ____“	Menampilkan “error”Pada hasil SearchResponse	Ditampilkan “Error”	Valid
2	Input data salah returnDate : “2019-09-19”	Menampilkan “errorMessage” Pada hasil SearchResponse	Ditampilkan “Error”	Valid

4.3.2 Pengujian Book

Pengujian terhadap fitur Book dilakukan pada sisi *developer* untuk menguji fungsi dan kinerja fitur. Pengujian fitur tersebut dilakukan dengan memberikan *input* pada *Book Request* sesuai dengan perancangan dan implementasi yang telah dibuat. Hasil pengujian dilihat berdasarkan hasil keluaran dari *Book Response* setelah mendapatkan dua jenis pengujian yaitu pengujian normal dan pengujian tidak normal. Adapun hasil pengujian fitur Book dapat dilihat pada Tabel 4.6.

Tabel 4.6 Pengujian Fitur Book

Pengujian Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Mengisi *Search Key* dengan benar “02080013877968808083796588877872868489767065687 2”	Menampilkan Hasil Book Response	Book Response ditampilkan	Valid
Pengujian Tidak Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Tidak mengisi atau isian salah pada *Search Key* “ “	Menampilkan “errorMessage” pada Book Response	Ditampilkan “errorMessage”: “Invalid Search Key”	Valid
2	Salah Mengisi *Search Key* “020800138779688080837 9”	Menampilkan “errorMessage” pada Book Response	Ditampilkan “errorMessage”: “Invalid Search Key”	Valid

4.3.3 Pengujian Payment

Pengujian terhadap fitur Payment dilakukan pada sisi *developer* untuk menguji fungsi dan kinerja fitur. Pengujian fitur tersebut dilakukan dengan memberikan *input* pada *Payment Request* sesuai dengan perancangan dan implementasi yang telah dibuat. Hasil pengujian dilihat berdasarkan hasil dari *Payment Response* setelah mendapatkan dua jenis pengujian yaitu pengujian normal dan pengujian tidak normal. Adapun hasil pengujian fitur Payment dapat dilihat pada Tabel 4.7.

Tabel 4.7 Pengujian Fitur Payment

Pengujian Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Mengisi “bookingCode” dengan benar “BKLJRJ”	Menampilkan hasil dari *Payment Response* berupa “paymentCode”	Payment Response menampilkan “paymentCode”	Valid

Pengujian Tidak Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Tidak mengisi “bookingCode” “___“	Menampilkan “errorMessage” Pada Payment Response	Ditampilkan “errorMessage”: “Generate Numeric PNR Failed 1”	Valid
2	Mengisi “bookingCode” dengan salah “BKLJ-RJ”	Menampilkan “errorMessage” Pada Payment Response	Ditampilkan “errorMessage”: “Generate Numeric PNR Failed 1”	Valid

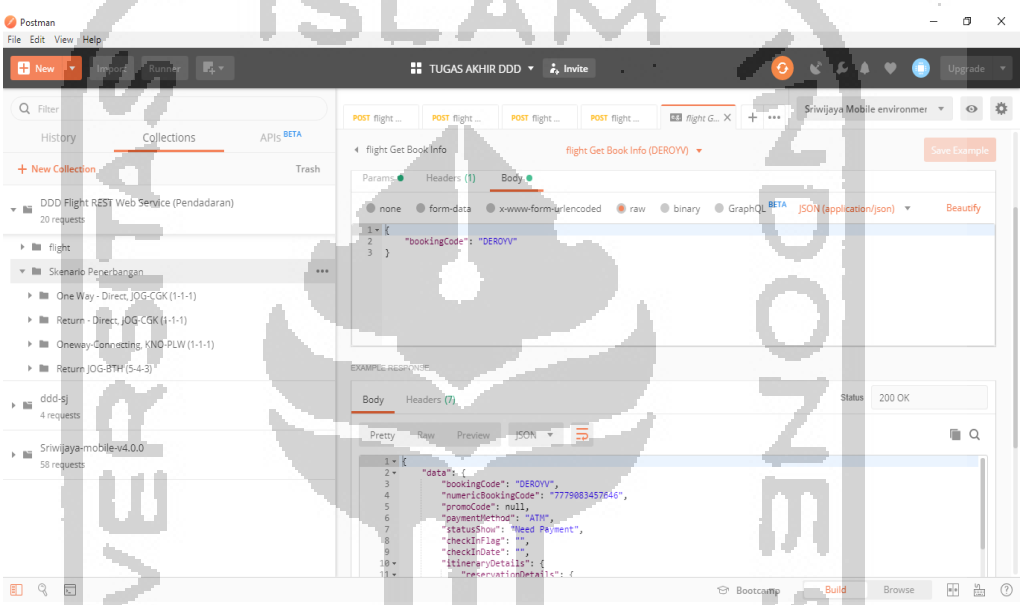
4.3.4 Pengujian Book Info

Pengujian terhadap fitur Book Info dilakukan pada sisi *developer* untuk menguji fungsi dan kinerja fitur. Pengujian fitur tersebut dilakukan dengan memberikan *input* pada *Book Info Request* sesuai dengan perancangan dan implementasi yang telah dibuat. Hasil pengujian dilihat berdasarkan hasil keluaran dari *Book Info Response* setelah mendapatkan dua jenis pengujian yaitu pengujian normal dan pengujian tidak normal. Adapun hasil pengujian fitur Payment dapat dilihat pada Tabel 4.8.

Tabel 4.8 Pengujian Fitur Book Info

Pengujian Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Mengisi “bookingCode” dengan benar “BKLJRJ”	Menampilkan hasil dari *Book Info Response* berupa detail data penerbangan.	Menampilkan hasil dari *Book Info Response* berupa detail “booking code”, “numericBookingCode”, dan informasi terkait penerbangan dan detail penumpang	Valid
Pengujian Tidak Normal				
No	**Data Request**	**Yang Diharapkan**	**Hasil Response**	**Kesimpulan**
1	Salah mengisikan “bookingCode” “BKLYY”	Menampilkan “errorMessage” pada *Book Info Response*	Ditampilkan “error message”: “Booking Code Not Found”	Valid
2	Mengosongkan isian :”bookingCode” “_____”	Menampilkan “errorMessage” pada *Book Info Response*	Ditampilkan “error message”: “Booking Code Not Found”	Valid

Hasil pengujian pada tabel-tabel diatas menunjukan bahwa pengujian 4 fitur dengan jumlah butir uji sebanyak 12 butir yang telah dibuat dan telah diuji dengan menggunakan metode *black box* dapat bejalan dengan baik, karena setelah dilakukan pengujian normal dan tidak normal sesuai dengan yang diharapkan maka kesimpulan yang dapat diambil adalah berhasil. Berikut adalah hasil ujicoba yang diperoleh dari 4 fitur REST Web Service dapat dilihat pada Gambar 4.2.



Gambar 4.2 Hasil Uji Coba dengan menggunakan Postman

Gambar 4.2 merupakan hasil yang didapatkan setelah dilakukan ujicoba dengan menggunakan Postman, hasil yang didapatkan dari rangkaian proses REST Web Service reservasi tiket pesawat terbang yang dimulai dengan Search untuk pencarian penerbangan, kemudian Book untuk proses pengisian detail penumpang, dilanjutkan dengan Payment untuk melakukan proses pembayaran dan didapatkan hasil melalui proses Get Book Info berupa kode booking.

Dari hasil ujicoba yang telah didapatkan maka terdapat beberapa masukan dari *developer* untuk pengembangan lebih lanjut, dengan mengimplementasikan konsep DDD yang belum diimplementasikan pada penelitian ini. Kemudian untuk meningkatkan fleksibilitas *domain* yang ada pada penelitian ini dapat di kembangkan dengan menambah *provider* API maskapai penerbangan yang lain.