1. Project Description:
2. Architecture
3. Business Entities
4. Data
5. Unit Testing
6. Screen Shots
7. Installation
8. **Project Description**

ManageEvents helps to identify the occupancy of seat over different events. There are multiple Events which happens regularly at different location. Each location organizes the event with specific seat count. When an event is organized, the seats are open for booking for the end customer.

ManageEvents Services help to identify how many seats are occupied/available at a time.

Below are some of the properties for Seats:

Available = T/F (True/False) – Booking Status in ManageEvents

Aisle = T/F (True/False) – isAisle Status in ManageEvents

Type = C/A (Child/Adult) – type Status in ManageEvents

1. **Architecture:**

Client  
( Browser/ Soap or Rest Client)

Server

Backend   
(POJO / Scalable Any DB)

1. **Business Entities:**

1

M

M

1

Event

Location

Seat

has

has

1. **Sample Data:**

ManageEvents gets data from a dao class which consists of a Java Object with sample data added. This class is scalable and can be used to connect any database (even using any ORM like Hibernate) or a Non-SQL database.

Sample data used is as under:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Event Id** | **Seat Type** | | **Is Aisle** | | **Booking Status** | | | | **Total Seats** | |
| **Seat Type** | | **Is Aisle** | |
| **A** | **C** | **T** | **F** | **A** | **C** | **T** | **F** | **Total** | **Booked** |
| 1 | 46 | 6 | 16 | 36 | 35 | 3 | 14 | 24 | **52** | **38** |
| 2 | 8 | 2 | 2 | 8 | 3 | 1 | 1 | 3 | **10** | **4** |
| 3 | 26 | 8 | 11 | 23 | 10 | 5 | 4 | 11 | **34** | **15** |

**Example:** For Event Id: 1, there are total 52 seats out of which 46 are Adults and 6 are Child. Similarly, out of 52 16 are Aisle and 36 are center seats. Total booked out of 52 is 38 and the event has 14 seats available for sell.

**Sample:**

[

{

"eventId": 1,

"totalSeats": 52,

"bookedSeats": 38

},

{

"eventId": 2,

"totalSeats": 10,

"bookedSeats": 4

},

{

"eventId": 3,

"totalSeats": 34,

"bookedSeats": 15

}

]

1. **Unit Testing:**

Note: The data set used here is the same as the one given in the Database/Backend Section

Business Scenarios Covered:

1. Total Count of Seats Available

Service: Get Service 🡪 getAvailSeats()

Input Parameters:

eventid 🡪 Integer >= 0

Test Scenarios:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr No | Test Case | Input Parameters | Result | Application Behavior |
| 1 | Validate Positive Case:  Passing Valid eventid  (http://localhost:8080/ManageEvents/rest/event?eventId=1) | eventid=1 | [  {  "eventId": 1,  "totalSeats": 52,  "bookedSeats": 38  }  ] | As Expected |
| 2 | Valid Positive Case:  Passing 0 to get all the events data  (http://localhost:8080/ManageEvents/rest/event?eventId=0) | eventid=0 | [  {  "eventId": 1,  "totalSeats": 52,  "bookedSeats": 38  },  {  "eventId": 2,  "totalSeats": 10,  "bookedSeats": 4  },  {  "eventId": 3,  "totalSeats": 34,  "bookedSeats": 15  }  ] | As Expected |
| 3 | Invalid Input  Passing non-numeric input  (http://localhost:8080/ManageEvents/rest/event?eventId=s) | eventid=s | The requested resource is not available | As Expected  Service not available for non-numeric eventid |
| 4 | Invalid Input  Passing negative number  (http://localhost:8080/ManageEvents/rest/event?eventId=-1) | eventid=-1 | Error Response |  |

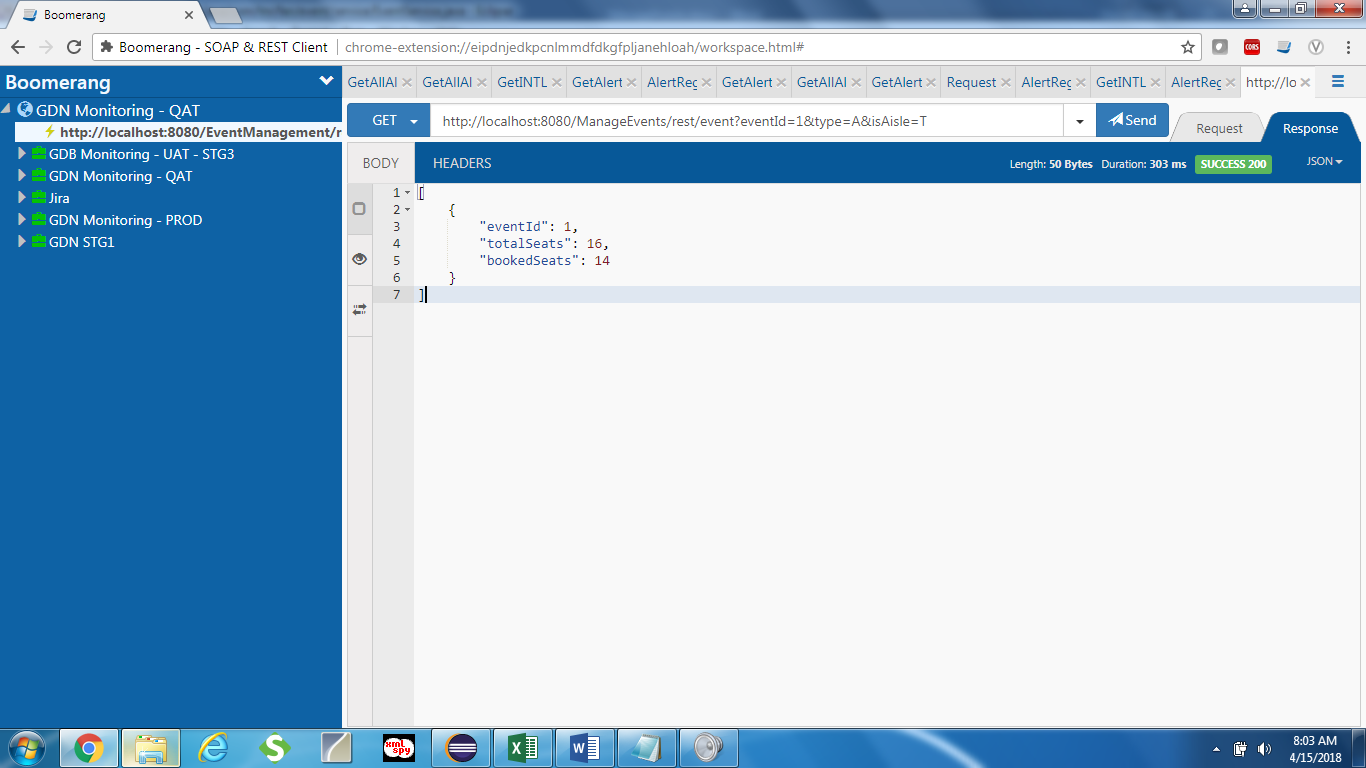
1. Seats based on Filter – Type

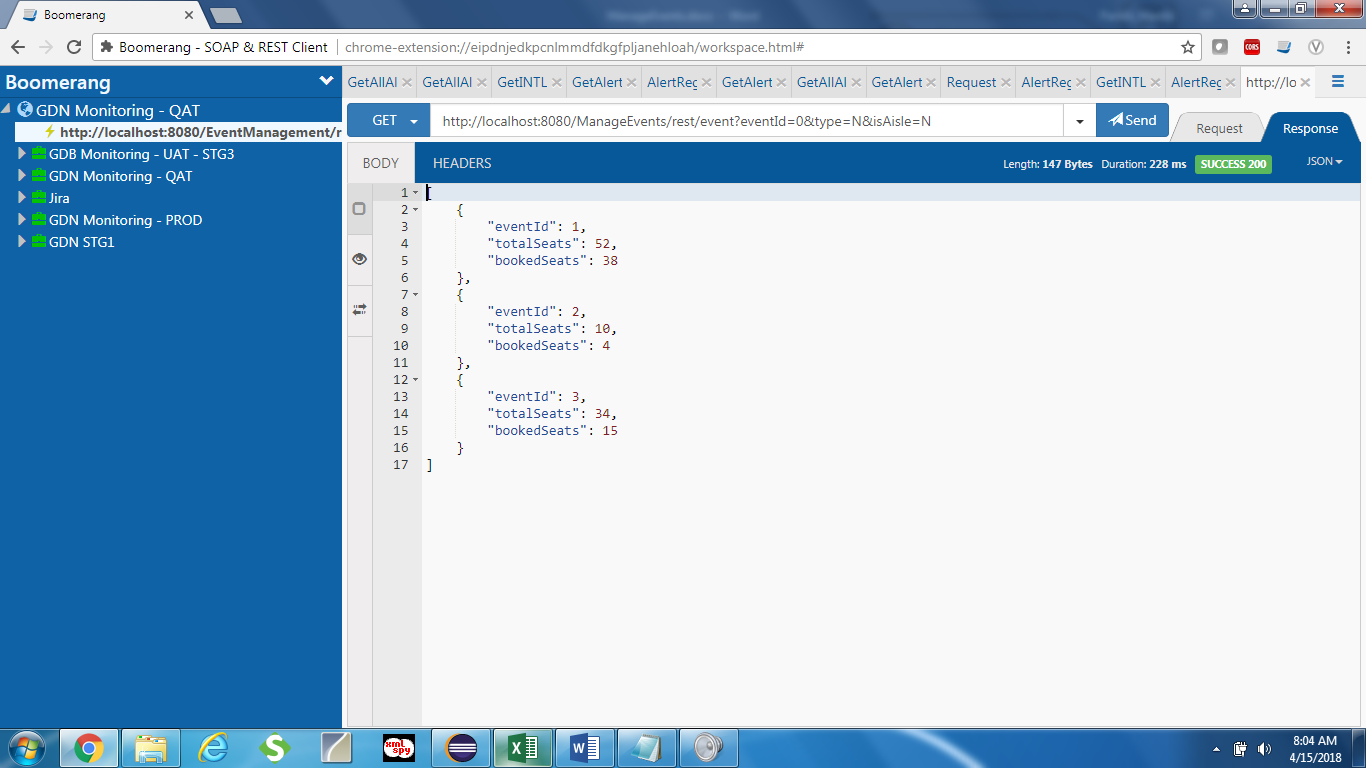
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr No | Test Case | Input Parameters | Result | Application Behavior |
| 1 | Validate Positive Case:  Passing Valid eventid and Adult Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=A) | eventid=1&type=A | [  {  "eventId": 1,  "totalSeats": 46,  "bookedSeats": 35  }  ] | As Expected |
| 2 | Validate Positive Case:  Passing Valid eventid and Child Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=C) | eventid=1&type=C | [  {  "eventId": 1,  "totalSeats": 6,  "bookedSeats": 3  }  ] | As Expected |
| 3 | Validate Positive Case:  Passing Valid eventid and no child type  (http://localhost:8080/ManageEvents/rest/event?eventId=1) | eventid=1 | Same as Section A test case number 1 | As Expected |
| 4 | Valid Positive Case:  Passing eventid as 0 and type as N to get all the events data  (http://localhost:8080/ManageEvents/rest/event?eventId=0&type=N) | eventid=0&type=N | [  {  "eventId": 1,  "totalSeats": 52,  "bookedSeats": 38  },  {  "eventId": 2,  "totalSeats": 10,  "bookedSeats": 4  },  {  "eventId": 3,  "totalSeats": 34,  "bookedSeats": 15  }  ] | As Expected |
| 5 | Valid Positive Case:  Passing eventid as 1 and type as N to get all the events data  (http://localhost:8080/ManageEvents/rest/event?eventId=0&type=N) | eventid=1&type=N | [  {  "eventId": 1,  "totalSeats": 52,  "bookedSeats": 38  }  ] | As Expected |
| 6 | Invalid Input  Passing numeric number or invalid Character for type  (<http://localhost:8080/ManageEvents/rest/event?eventId=1>&type=1)  OR  (<http://localhost:8080/ManageEvents/rest/event?eventId=s>&type=s) | eventid=1&type=1  OR  eventid=1&type=s | Error Response | As Expected |

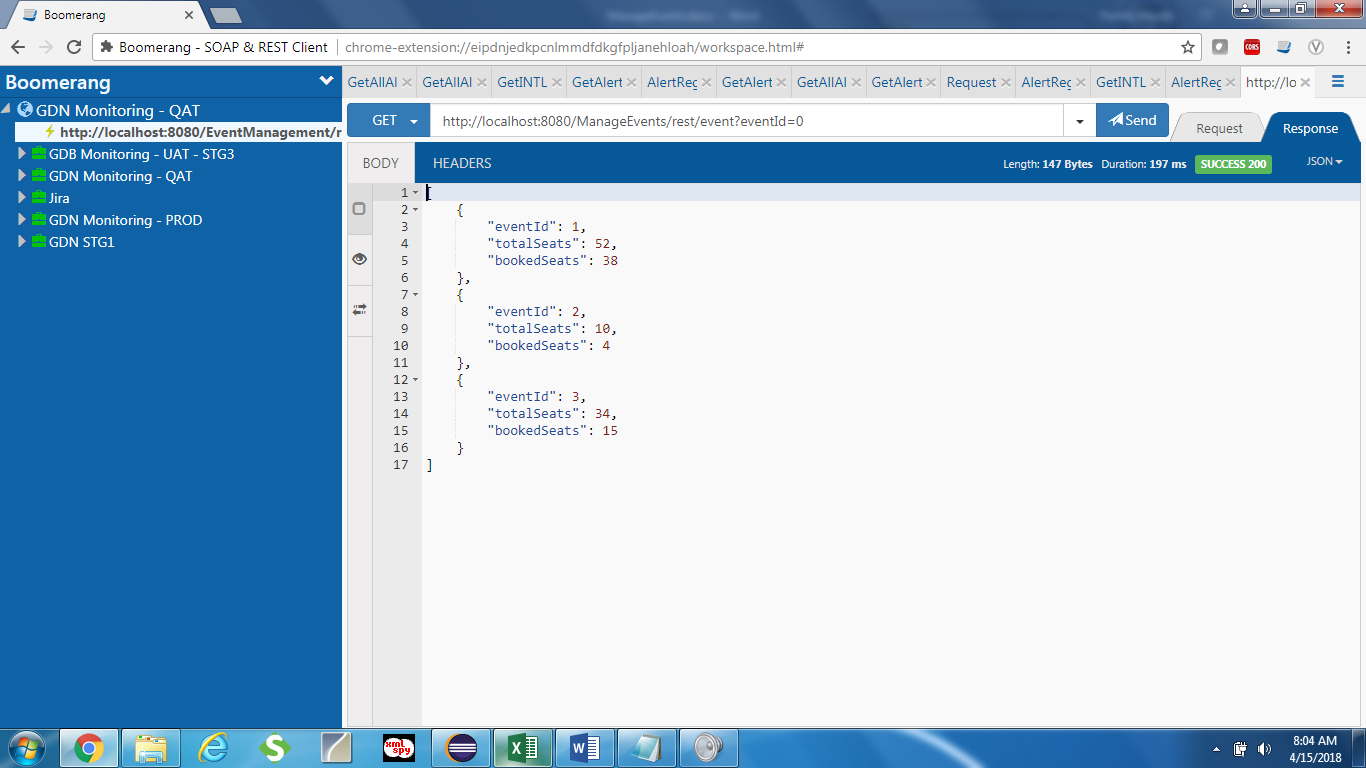
1. Seat based on Filter – Aisle

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sr No | Test Case | Input Parameters | Result | Application Behavior |
| 1 | Validate Positive Case:  Passing Valid eventid and Adult Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=A&isAisle=T) | eventid=1&type=A&isAisle=T | [  {  "eventId": 1,  "totalSeats": 16,  "bookedSeats": 14  }  ] | As Expected |
| 2 | Validate Positive Case:  Passing Valid eventid and Child Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=C&isAisle=T) | eventid=1&type=C&isAisle=T | [  {  "eventId": 1,  "totalSeats": 0,  "bookedSeats": 0  }  ] | As Expected |
| 3 | Validate Positive Case:  Passing Valid eventid and Adult Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=A&isAisle=F) | eventid=1&type=A&isAisle=F | [  {  "eventId": 1,  "totalSeats": 30,  "bookedSeats": 21  }  ] | As Expected |
| 4 | Validate Positive Case:  Passing Valid eventid and Child Type  (http://localhost:8080/ManageEvents/rest/event?eventId=1&type=C&isAisle=F) | eventid=1&type=C&isAisle=F | [  {  "eventId": 1,  "totalSeats": 6,  "bookedSeats": 3  }  ] | As Expected |
| 5 | Valid Positive Case:  Passing eventid as 0 and type as N to get all the events data  (http://localhost:8080/ManageEvents/rest/event?eventId=0&type=N&isAisle=N) | eventid=0&type=N&isAisle=N | [  {  "eventId": 1,  "totalSeats": 52,  "bookedSeats": 38  },  {  "eventId": 2,  "totalSeats": 10,  "bookedSeats": 4  },  {  "eventId": 3,  "totalSeats": 34,  "bookedSeats": 15  }  ] | As Expected |
| 6 | Invalid Input  Passing numeric number or invalid Character for type  (<http://localhost:8080/ManageEvents/rest/event?eventId=1>&type=A&isAisle=-1)  OR  (<http://localhost:8080/ManageEvents/rest/event?eventId=s>&type=A&isAisle=s) | eventid=1&type=A&isAisle=-1  OR  eventid=1&type=A&isAisle=s | Error Response | As Expected |

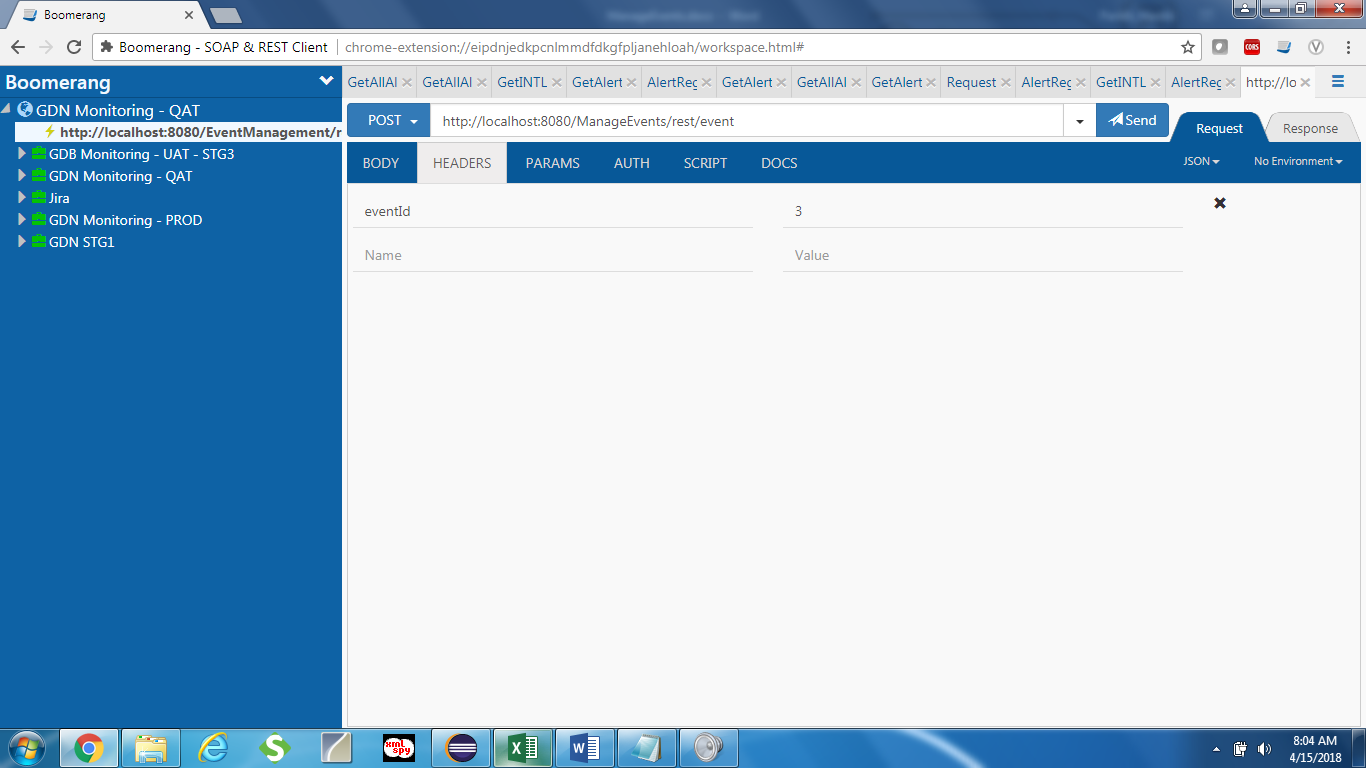
1. **Screen Shots:**

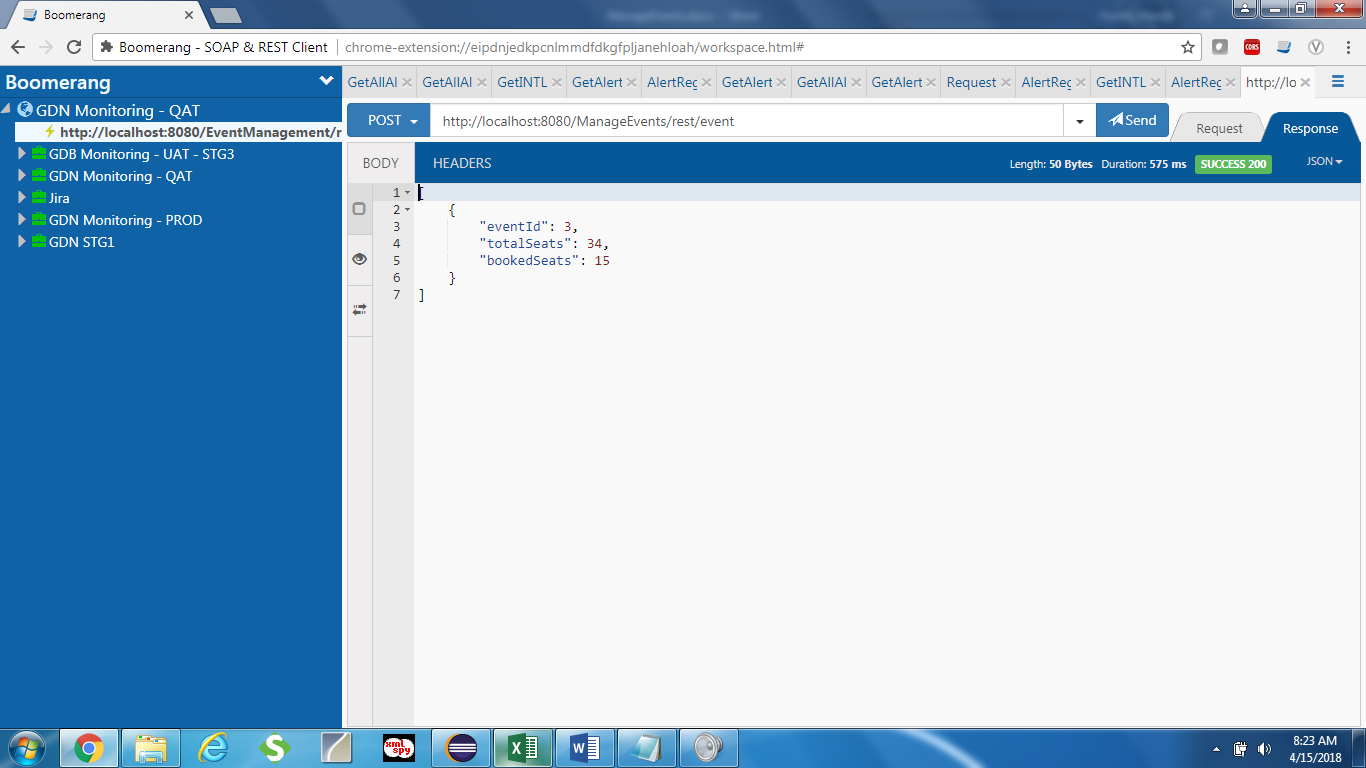






**Post Method: Fetch - AllEventDetails**





1. **Installation:**

**Git Repo URL:** <https://github.com/HardikParikh/ManageEvents>

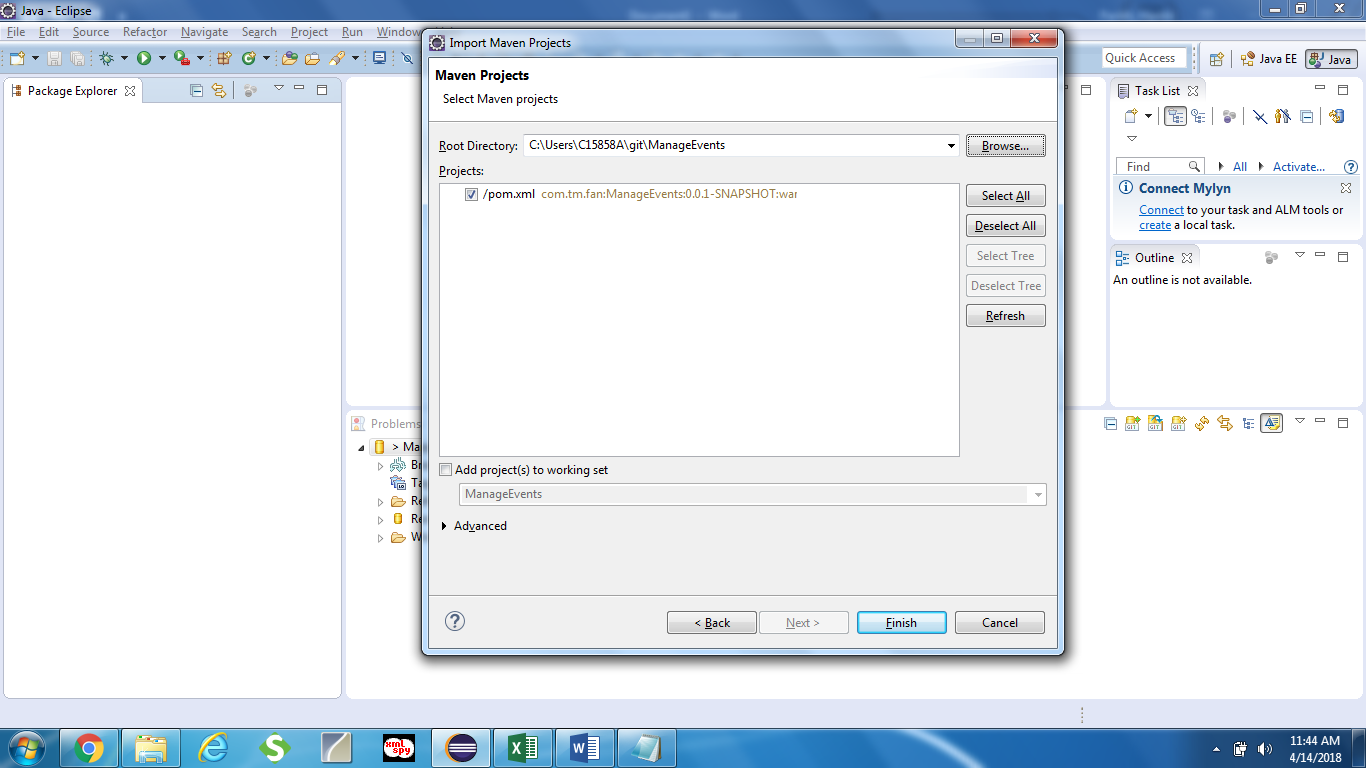
**Step1:** Open Git Repositories View and provide the GitHub Project Details and Select Master Branch

|  |  |
| --- | --- |
|  |  |

Step 2: Import Project and Select Existing Maven Option

|  |  |
| --- | --- |
|  |  |

Step 3: Select File and Click Finish.



Step 4: Update Maven Project and Clean Install on the Server to deploy