5COSC019W

Object Oriented Programming

Test Report

Name: Orthalange Tharushi Madubhashinie

Student Number: 20231033

UoW number: w2051854

Tutorial Group: 08(SE)

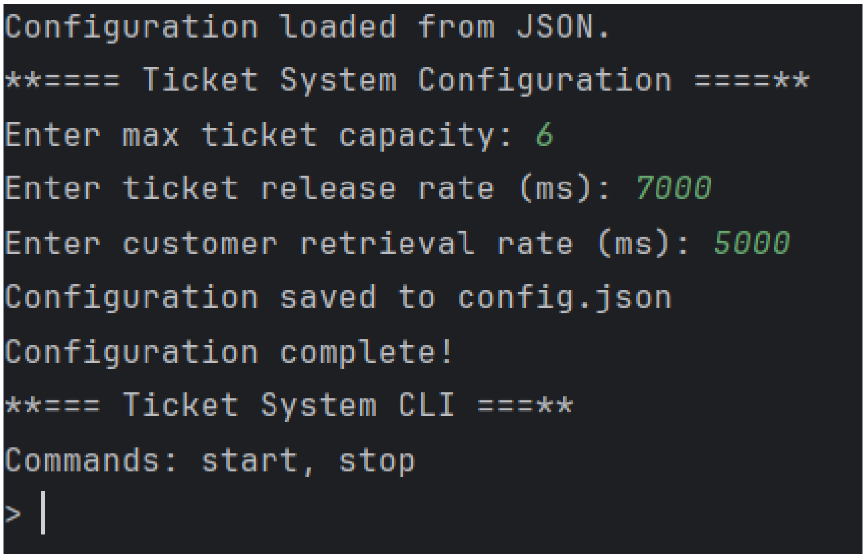
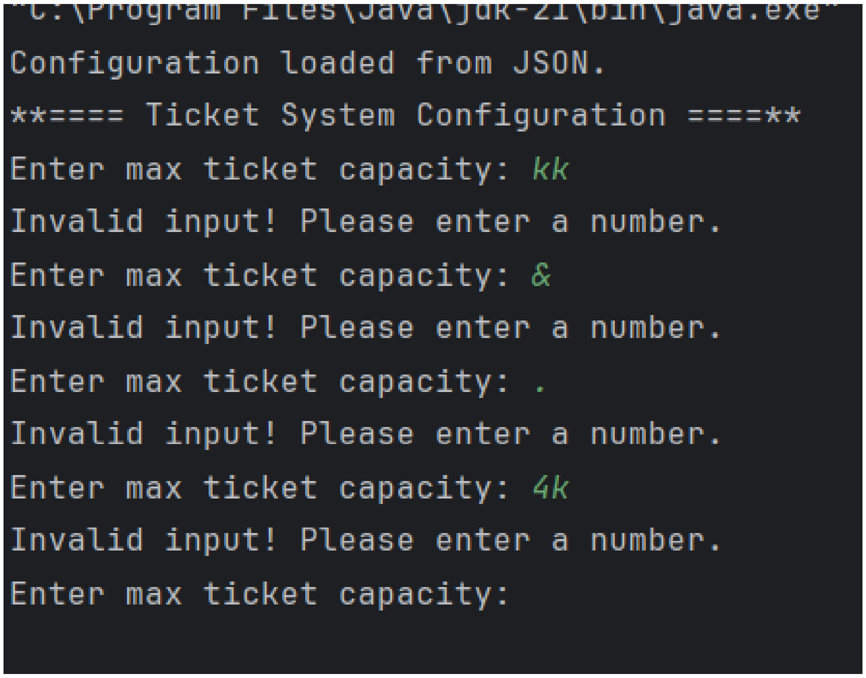
Contents

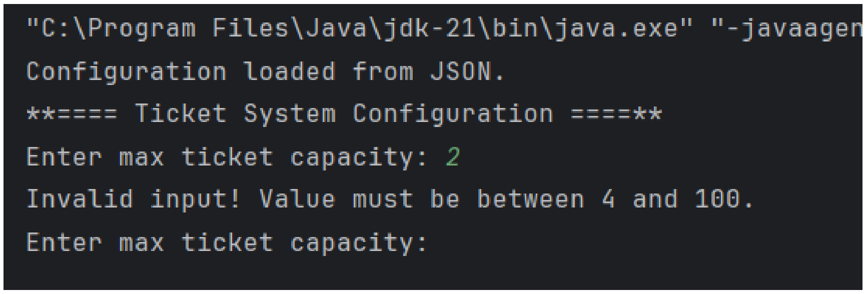
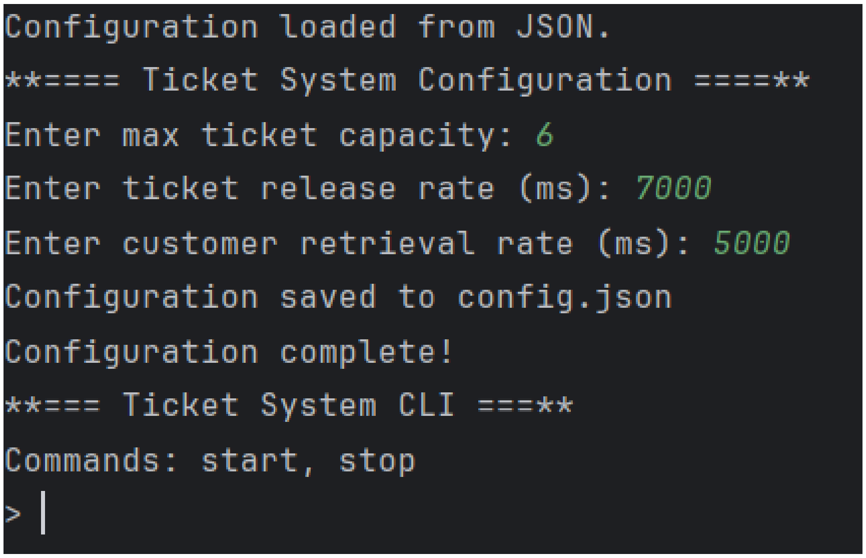
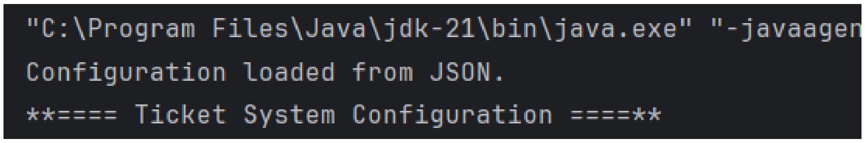
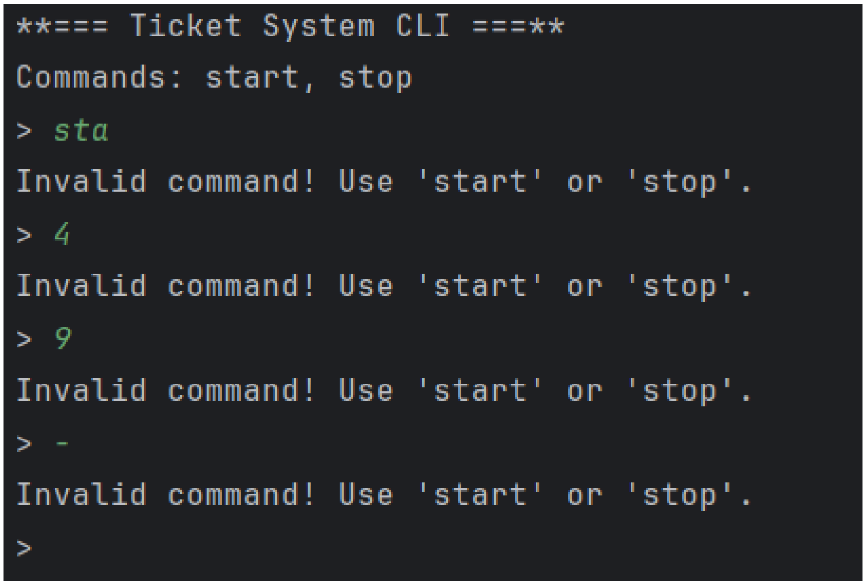
CLI PART

Test Case Number	Test Case	Expected Output	Actual Output	Pass / Fail
1	Validate user input within the range	Correct value is returned and displays “Configuration complete!”	Correct value is returned and displays “Configuration complete!	Pass
2	Enter invalid input	Displays “Invalid input! Please enter a number.” and prompts again	Displays “Invalid input! Please enter a number.” and prompts again	Pass
3	Validate user input is not within range	Displays “Invalid input! Value must be between a minimum value and maximum value”	Displays “Invalid input! Value must be between a minimum value and maximum value”	Pass
4	Save configurations to JSON	JSON file created and saved configurations. Displays “Configuration saved to file_name”	JSON file created and saved configurations. Displays “Configuration saved to file_name”	Pass
5	Fail to save configurations to JSON	Display “Error saving configuration, the error message”	Display “Error saving configuration, the error message”	Pass
6	Load configurations from JSON file	Data loaded from file and Display “Configuration loaded from JSON”. If it fails to load data, display “Error reading configuration”	Data loaded from file and Display “Configuration loaded from JSON”. If it fails to load data, display “Error reading configuration”	Pass

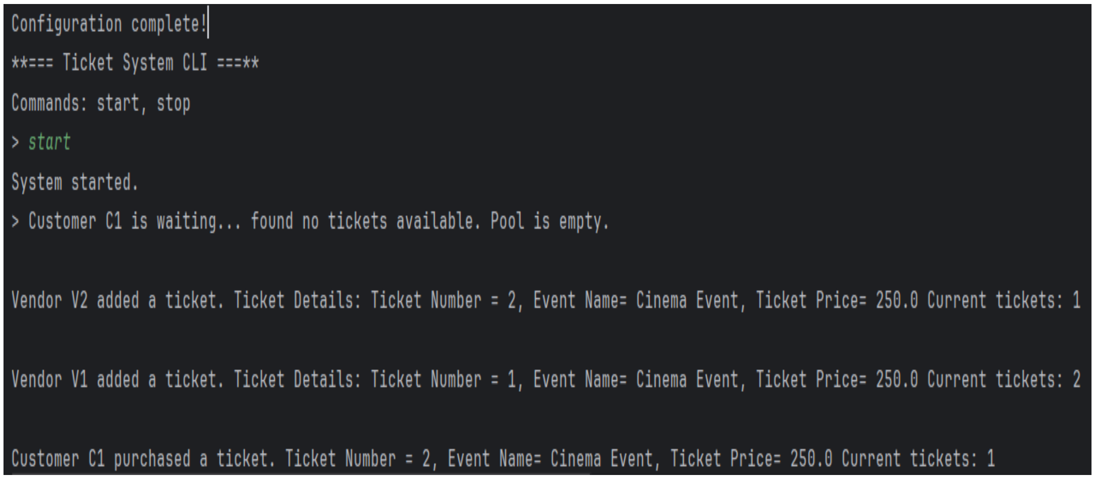
7	Enter Invalid input for start	Display “Invalid command! Use 'start' or 'stop'” and prompt again	Display “Invalid command! Use 'start' or 'stop'” and prompt again	Pass
8	Enter “Start”	Run the threading part and display “System started.”	Run the threading part and display “System started.”	Pass
9	Enter Invalid input for stop	Display “Invalid command! Use 'start' or 'stop'”	Display “Invalid command! Use 'start' or 'stop'”	Pass
10	Enter “Stop”	Stop system and display “System stopped. Thanks for using our Ticket Management System! “. and save configurations to config.json	Stop system and display “System stopped. Thanks for using our Ticket Management System! “. and save configurations to config.json	Pass

Appendix for CLI part

Test Case Number	Picture
1	 Configuration loaded from JSON. **===== Ticket System Configuration ====** Enter max ticket capacity: 6 Enter ticket release rate (ms): 7000 Enter customer retrieval rate (ms): 5000 Configuration saved to config.json Configuration complete! **=== Ticket System CLI ===** Commands: start, stop > |
2	 "C:\Program Files\Java\jdk-21\bin\java.exe" Configuration loaded from JSON. **===== Ticket System Configuration ====** Enter max ticket capacity: kk Invalid input! Please enter a number. Enter max ticket capacity: & Invalid input! Please enter a number. Enter max ticket capacity: . Invalid input! Please enter a number. Enter max ticket capacity: 4k Invalid input! Please enter a number. Enter max ticket capacity:

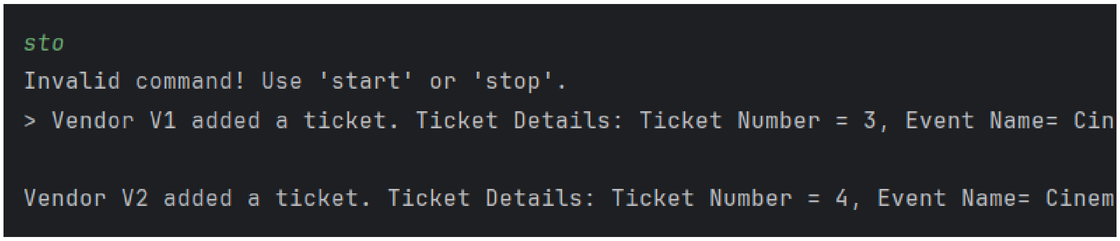
3	 "C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagen Configuration loaded from JSON. **===== Ticket System Configuration ====** Enter max ticket capacity: 2 Invalid input! Value must be between 4 and 100. Enter max ticket capacity:
4	 Configuration loaded from JSON. **===== Ticket System Configuration ====** Enter max ticket capacity: 6 Enter ticket release rate (ms): 7000 Enter customer retrieval rate (ms): 5000 Configuration saved to config.json Configuration complete! **=== Ticket System CLI ===** Commands: start, stop >
6	 "C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagen Configuration loaded from JSON. **===== Ticket System Configuration ====**
7	 **=== Ticket System CLI ===** Commands: start, stop > sta Invalid command! Use 'start' or 'stop'. > 4 Invalid command! Use 'start' or 'stop'. > 9 Invalid command! Use 'start' or 'stop'. > - Invalid command! Use 'start' or 'stop'. >

8



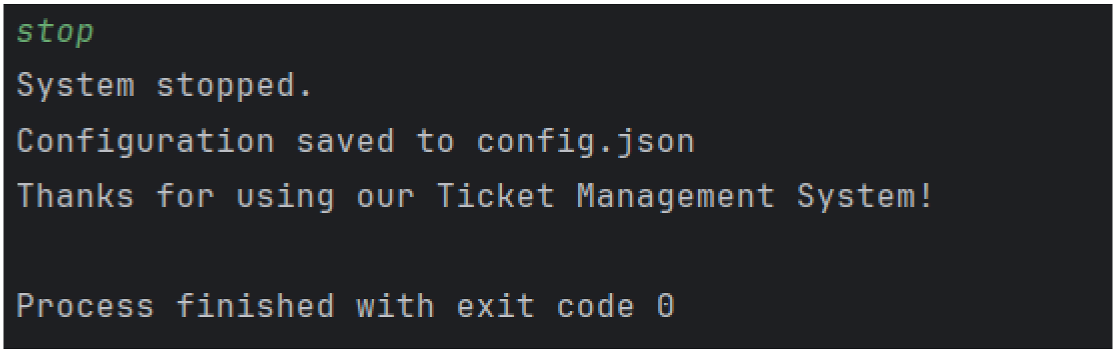
```
Configuration complete!
**=== Ticket System CLI ===**
Commands: start, stop
> start
System started.
> Customer C1 is waiting... found no tickets available. Pool is empty.

Vendor V2 added a ticket. Ticket Details: Ticket Number = 2, Event Name= Cinema Event, Ticket Price= 250.0 Current tickets: 1

Vendor V1 added a ticket. Ticket Details: Ticket Number = 1, Event Name= Cinema Event, Ticket Price= 250.0 Current tickets: 2

Customer C1 purchased a ticket. Ticket Number = 2, Event Name= Cinema Event, Ticket Price= 250.0 Current tickets: 1
```

9



```
sto
Invalid command! Use 'start' or 'stop'.
> Vendor V1 added a ticket. Ticket Details: Ticket Number = 3, Event Name= Cin

Vendor V2 added a ticket. Ticket Details: Ticket Number = 4, Event Name= Cinem
```

10



```
stop
System stopped.
Configuration saved to config.json
Thanks for using our Ticket Management System!

Process finished with exit code 0
```

Frontend And Backend

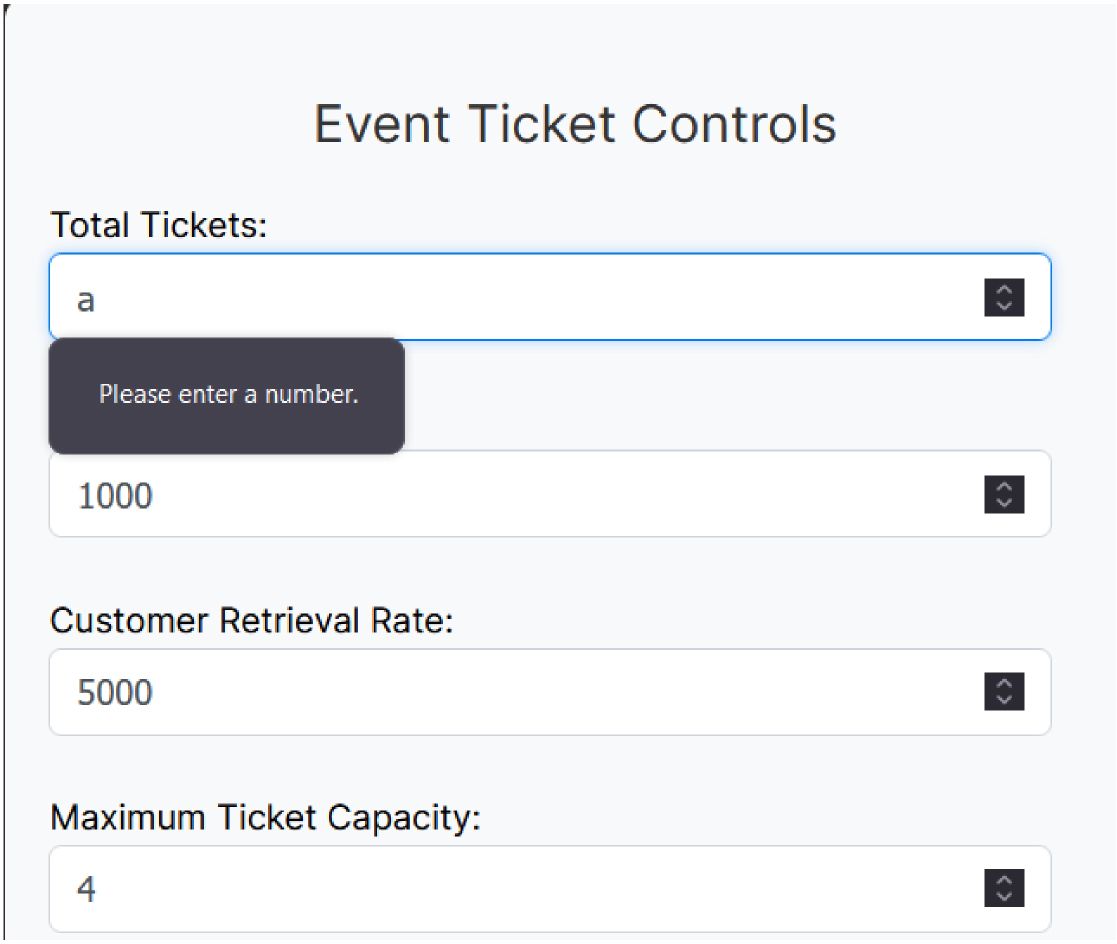
Test Case Number	Test Case	Expected Output	Actual Output	Pass / Fail
1	Enter invalid input	Display “Please enter a number”	Display “Please enter a number”	Pass
2	Maximum ticket capacity is not within the range	Display “Max Ticket Capacity must be between minimum value and maximum value.”	Display “Max Ticket Capacity must be between minimum value and maximum value.”	Pass
3	Failed to send configurations to the backend	Display “Failed to update configuration.”	Display “Failed to update configuration.”	Pass
4	Upload configurations to the backend	Display “Configurations saved successfully.” And display “Configuration complete!” in the console.	Display “Configurations saved successfully.” And display “Configuration complete!” in the console.	Pass
5	Press start button	Disable the start button and active stop button. Display “Ticket pool started.” Start running threads in the backend.	Disable the start button and active stop button. Display “Ticket pool started.” Start running threads in the backend.	Pass
6	Press reset button	Disable stop button and	Disable stop button and	Pass

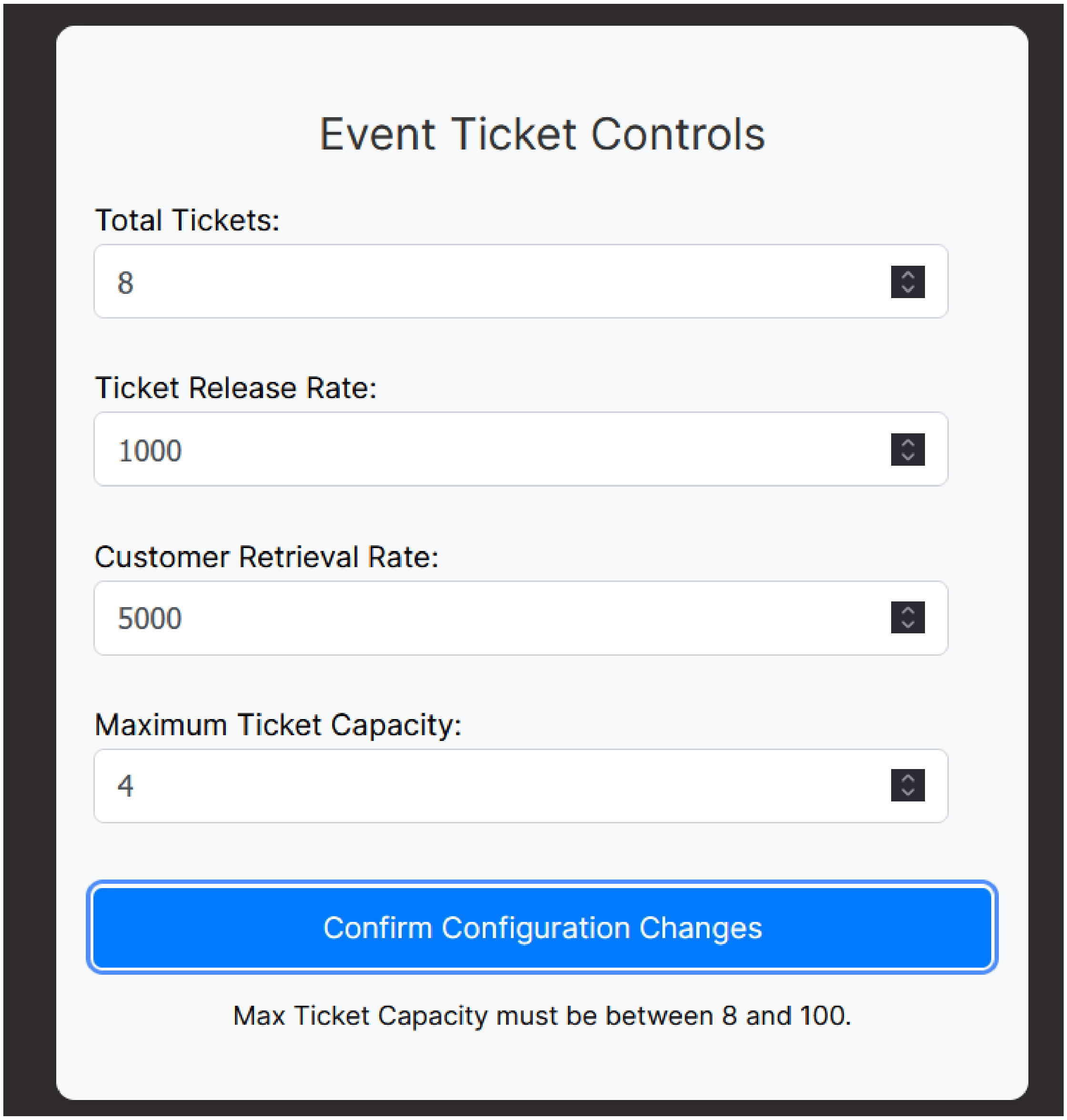
		active start button. Display “Ticket pool reset.” Keep running threads in the backend.	active start button. Display “Ticket pool reset.” Keep running threads in the backend.	
7	Press stop button	Disable the stop button, active start button, and reset button. Display “Ticket pool stopped.” Stop running threads in the backend.	Disable the stop button, active start button, and reset button. Display “Ticket pool stopped.” Stop running threads in the backend.	Pass
8	Failed to load details from the backend in the Ticket Pool status section	Display “Failed to load ticket status.”	Display “Failed to load ticket status.”	Pass
9	Successfully load details from the backend in the Ticket Pool status section	Display Remaining tickets in the current thread count and maximum ticket capacity count	Display Remaining tickets in the current thread count and maximum ticket capacity count	Pass
10	Handle empty log display	Display “No transactions available”	Display “No transactions available”	Pass
11	Display transactions in the log display	Display Transaction details in the table and display transaction details in the console.	Display Transaction details in the table and display transaction details in the console.	Pass
12	Store configurations	Create a	Create a	Pass

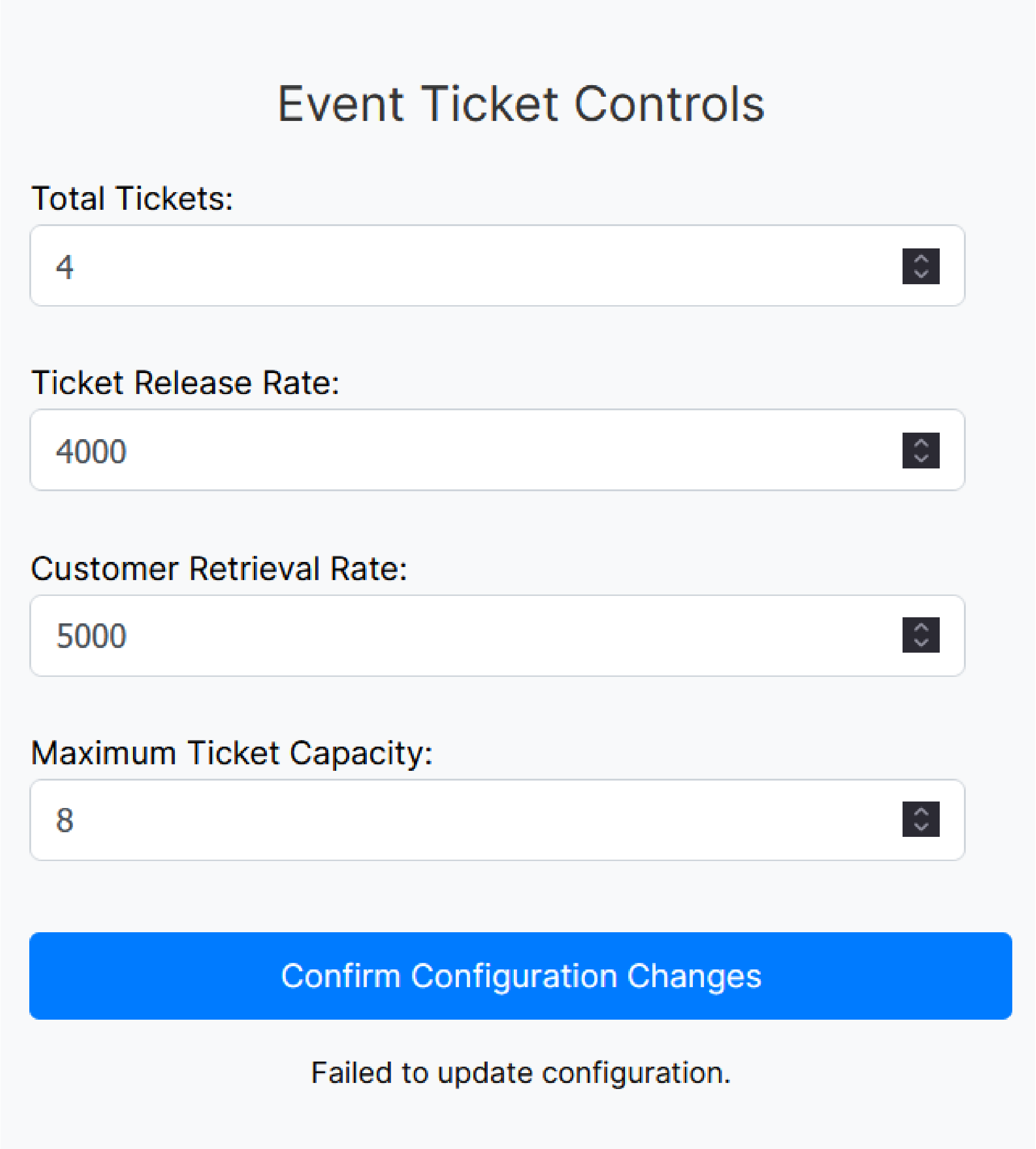
	in the Database	“configuration “table and update data.	“configuration “table and update data.	
13	Store tickets in the Database	Create a “ticket “table and update data.	Create a “ticket “table and update data.	Pass
14	Store transactions in the Database	Create a “transactions “table and update data.	Create a “transactions “table and update data.	Pass
15	Handle Configuration Rest API	200 OK	200 OK	Pass
16	Handle Control Panel Rest API	200 OK	200 OK	Pass
17	Handle Ticket Rest API	200 OK	200 OK	Pass
18	Handle Transaction Rest API	200 OK	200 OK	Pass

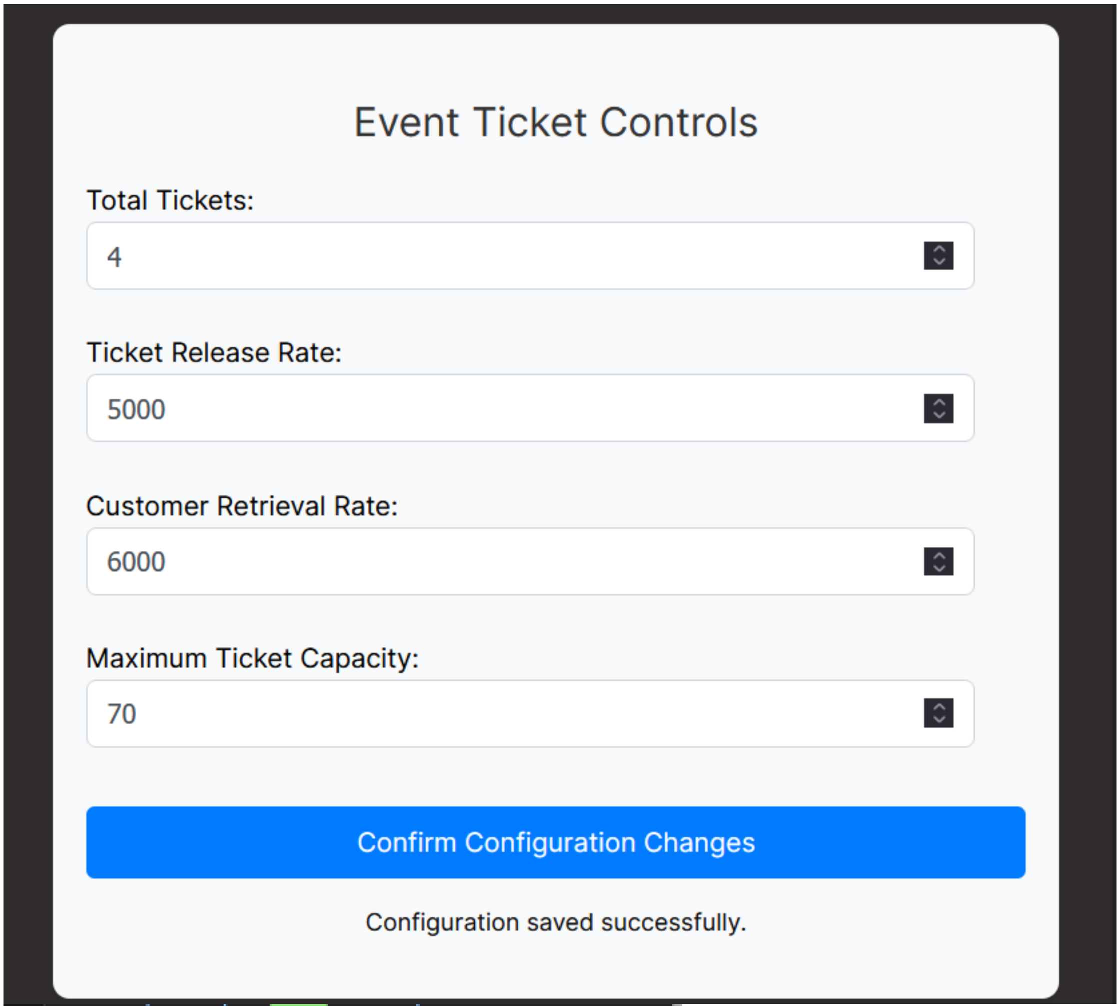
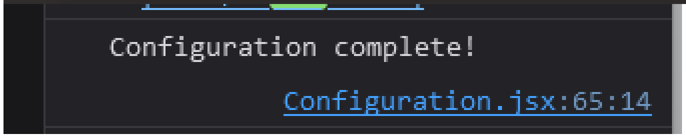
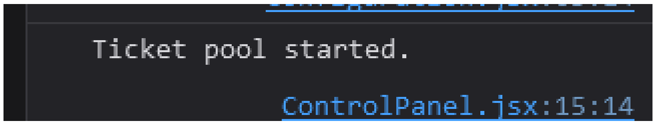
Appendix for Frontend and Backend

Test Case Number	Picture

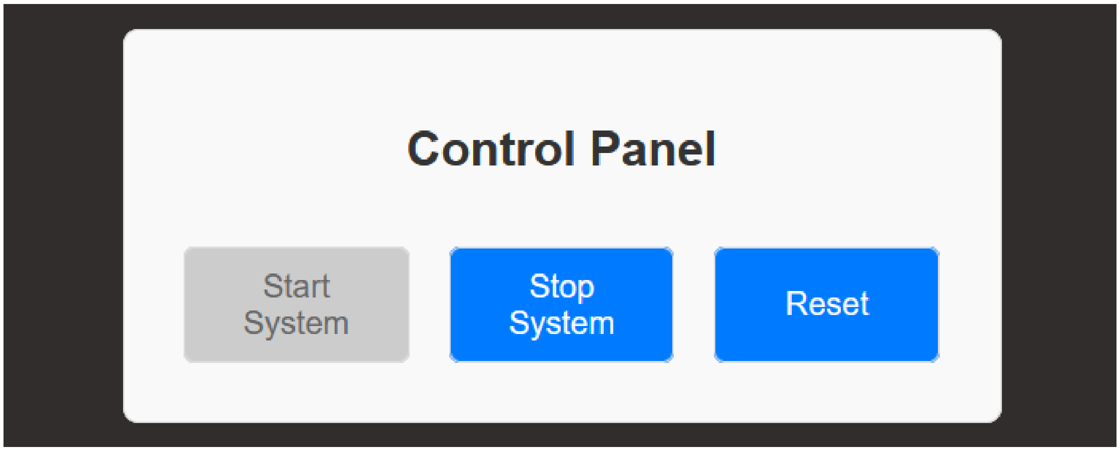
1	 Event Ticket Controls Total Tickets: a Please enter a number. 1000 Customer Retrieval Rate: 5000 Maximum Ticket Capacity: 4

2	 Event Ticket Controls Total Tickets: 8 Ticket Release Rate: 1000 Customer Retrieval Rate: 5000 Maximum Ticket Capacity: 4 Max Ticket Capacity must be between 8 and 100.

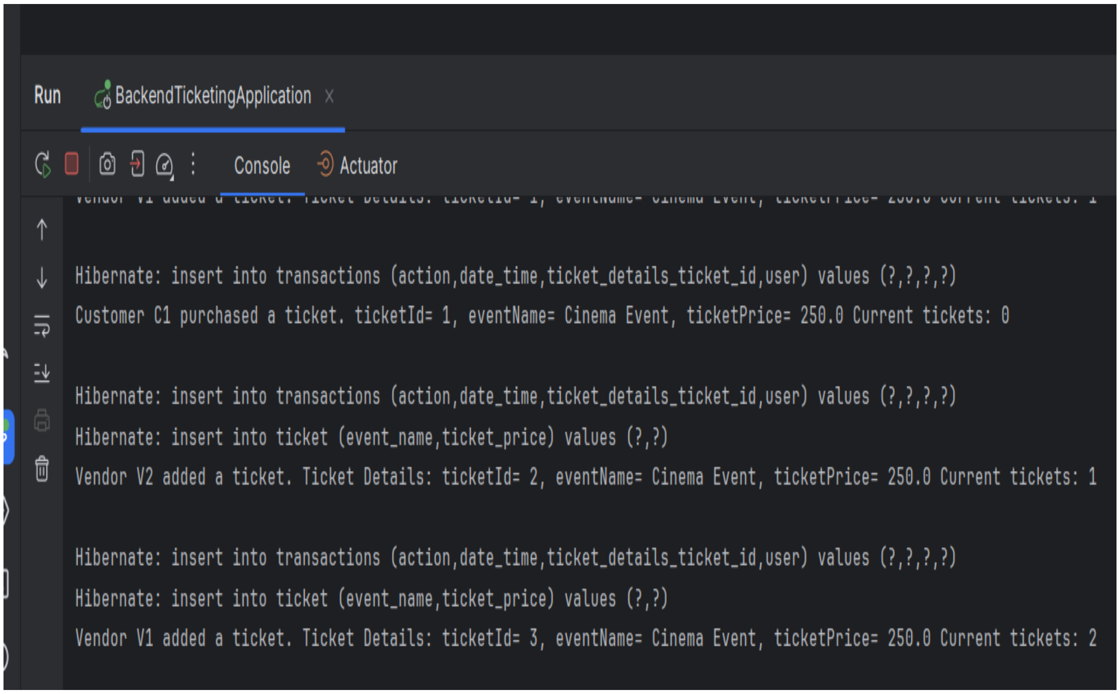
3	 Event Ticket Controls Total Tickets: 4 Ticket Release Rate: 4000 Customer Retrieval Rate: 5000 Maximum Ticket Capacity: 8 Failed to update configuration.

4	 Event Ticket Controls Total Tickets: 4 Ticket Release Rate: 5000 Customer Retrieval Rate: 6000 Maximum Ticket Capacity: 70 Configuration saved successfully.  Configuration complete! Configuration.jsx:65:14
5	 Ticket pool started. ControlPanel.jsx:15:14

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

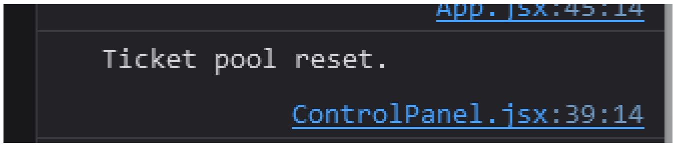


</td>
</tr>
<tr>
<td></td>
<td>



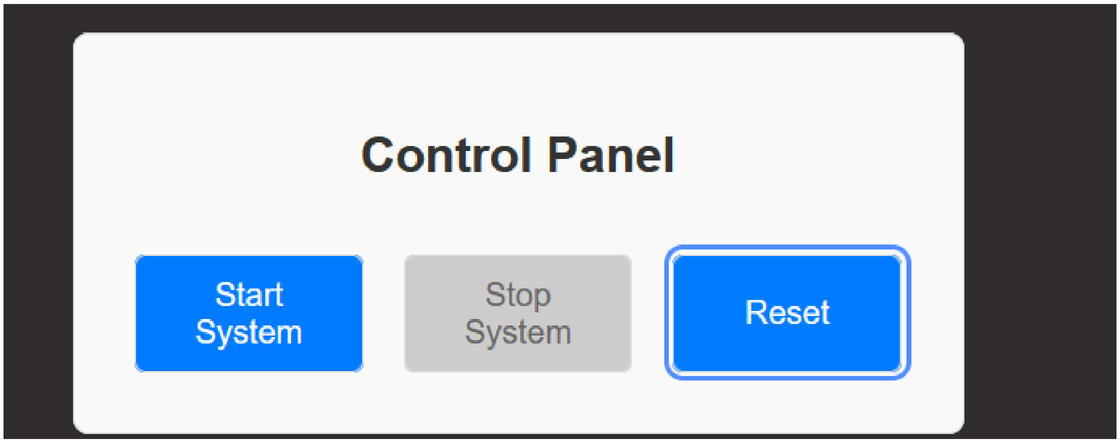
```
Vendor V1 added a ticket. Ticket Details: ticketId= 1, eventName= Cinema Event, ticketPrice= 250.0 Current tickets: 1

Hibernate: insert into transactions (action,date_time,ticket_details_ticket_id,user) values (?,?,?,?)
Customer C1 purchased a ticket. ticketId= 1, eventName= Cinema Event, ticketPrice= 250.0 Current tickets: 0

Hibernate: insert into transactions (action,date_time,ticket_details_ticket_id,user) values (?,?,?,?)
Hibernate: insert into ticket (event_name,ticket_price) values (?,?)
Vendor V2 added a ticket. Ticket Details: ticketId= 2, eventName= Cinema Event, ticketPrice= 250.0 Current tickets: 1

Hibernate: insert into transactions (action,date_time,ticket_details_ticket_id,user) values (?,?,?,?)
Hibernate: insert into ticket (event_name,ticket_price) values (?,?)
Vendor V1 added a ticket. Ticket Details: ticketId= 3, eventName= Cinema Event, ticketPrice= 250.0 Current tickets: 2
```

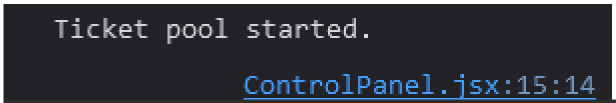
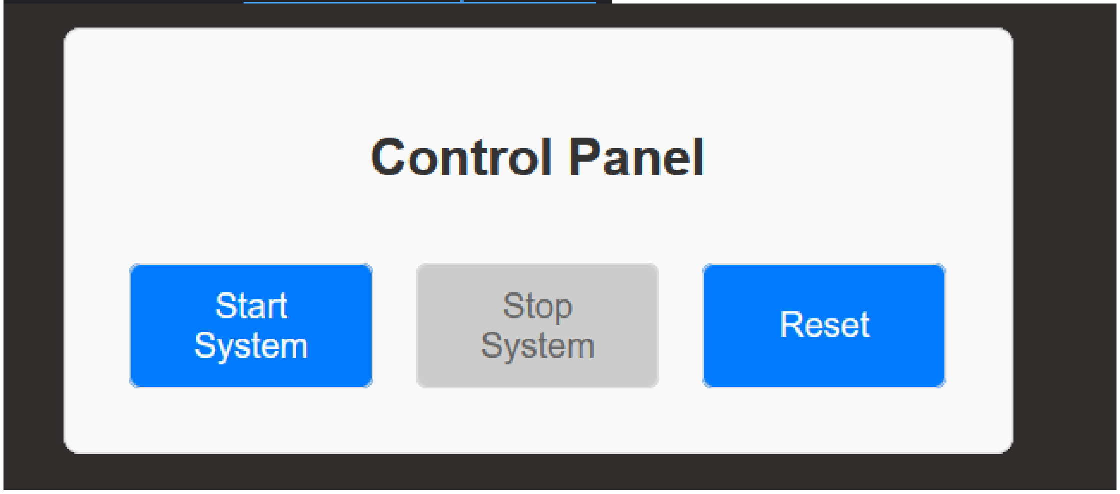
</td>
</tr>
<tr>
<td>6</td>
<td>

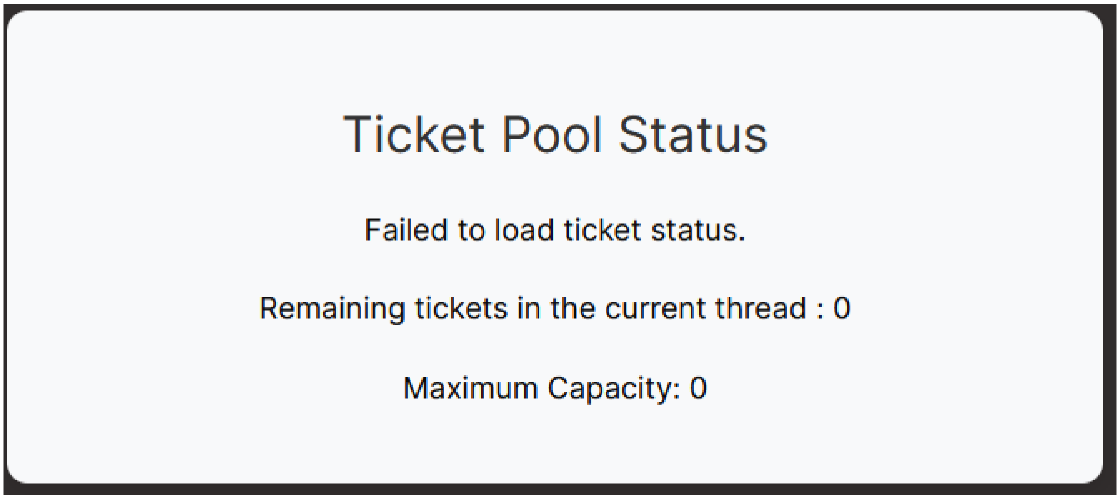


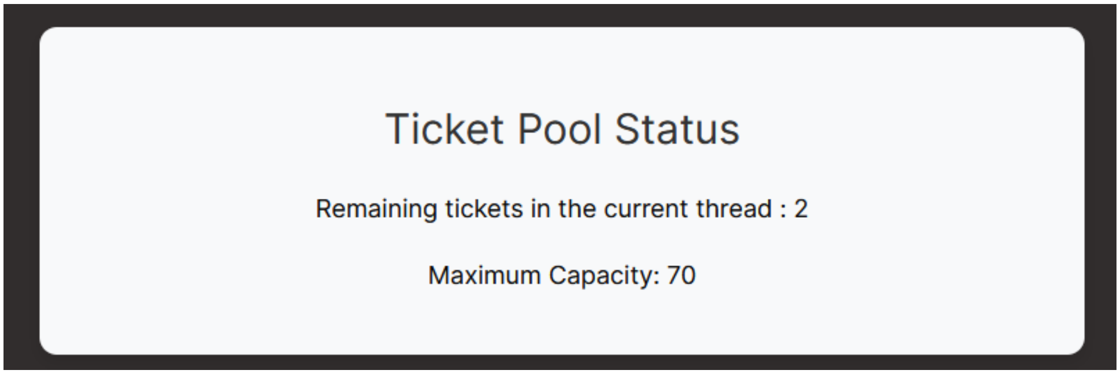
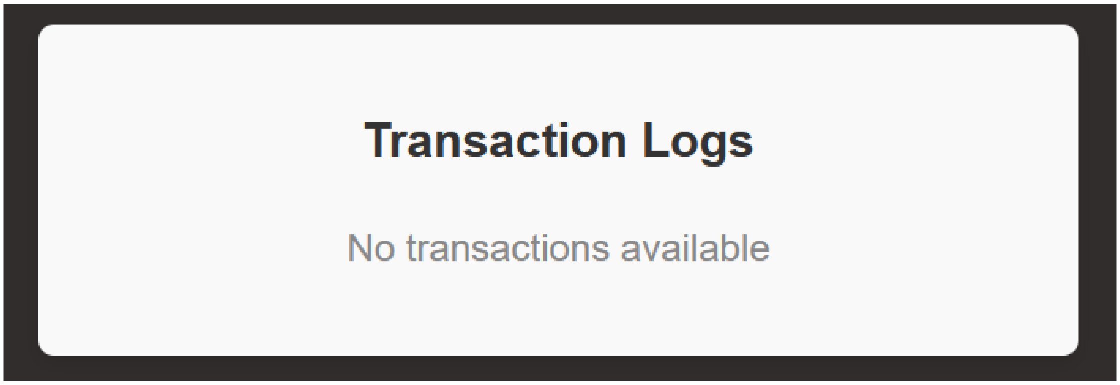
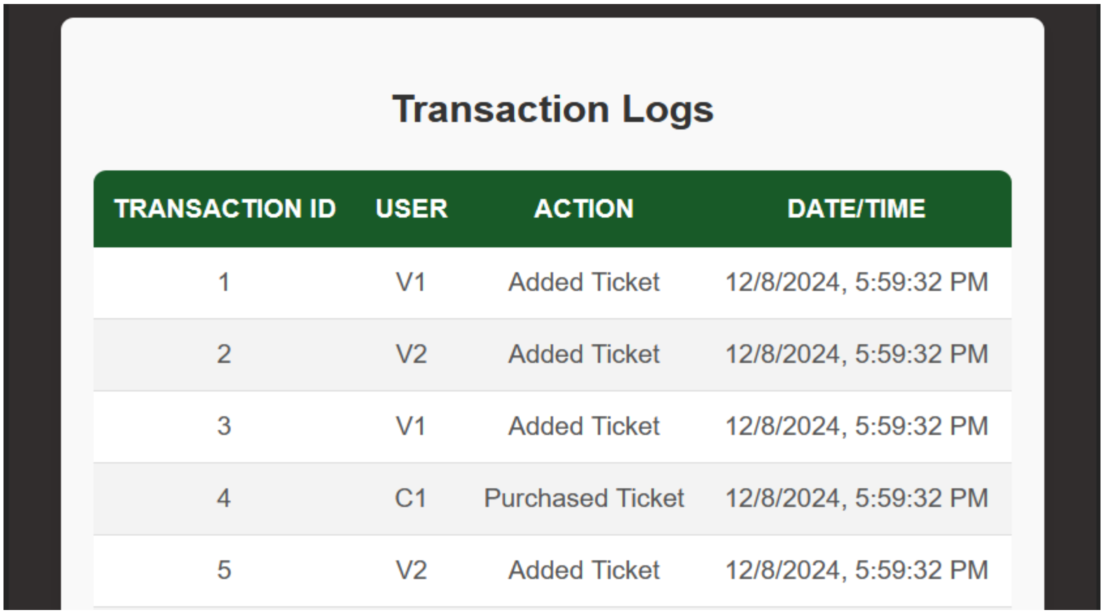
```
                                   App.jsx:45:14
  Ticket pool reset.
                           ControlPanel.jsx:39:14
```

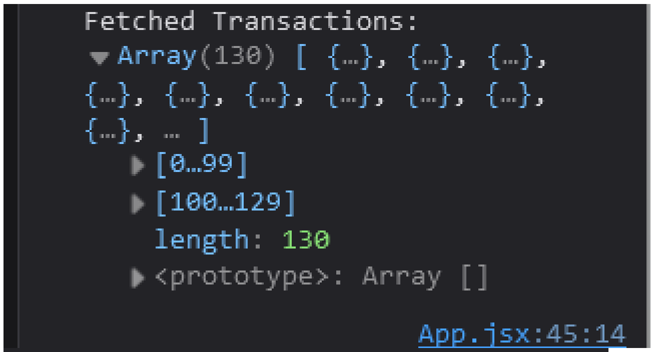
</td>
</tr>
</table>

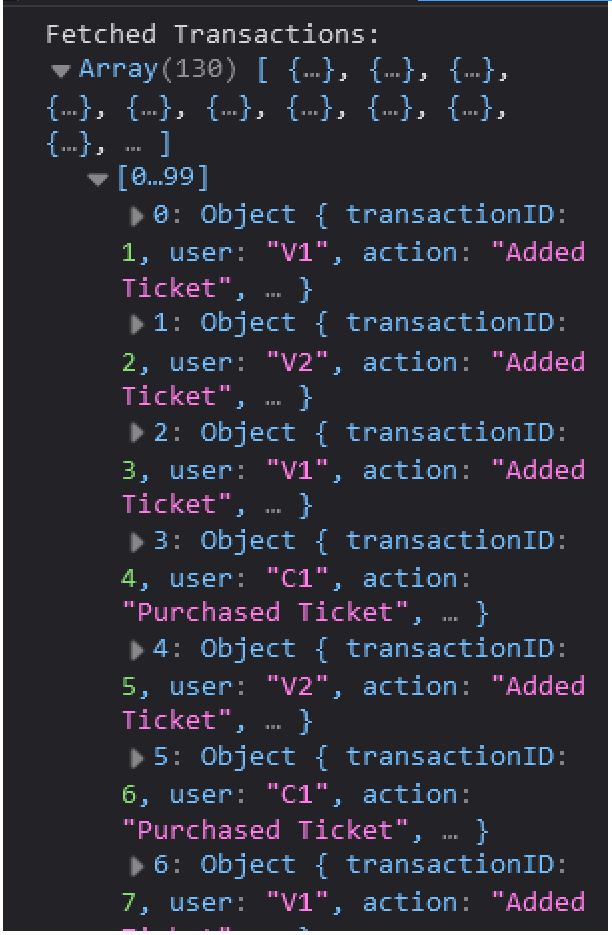
7	
	`Ticket pool started.`
	`ControlPanel.jsx:15:14`
	

8	 Ticket Pool Status Failed to load ticket status. Remaining tickets in the current thread : 0 Maximum Capacity: 0

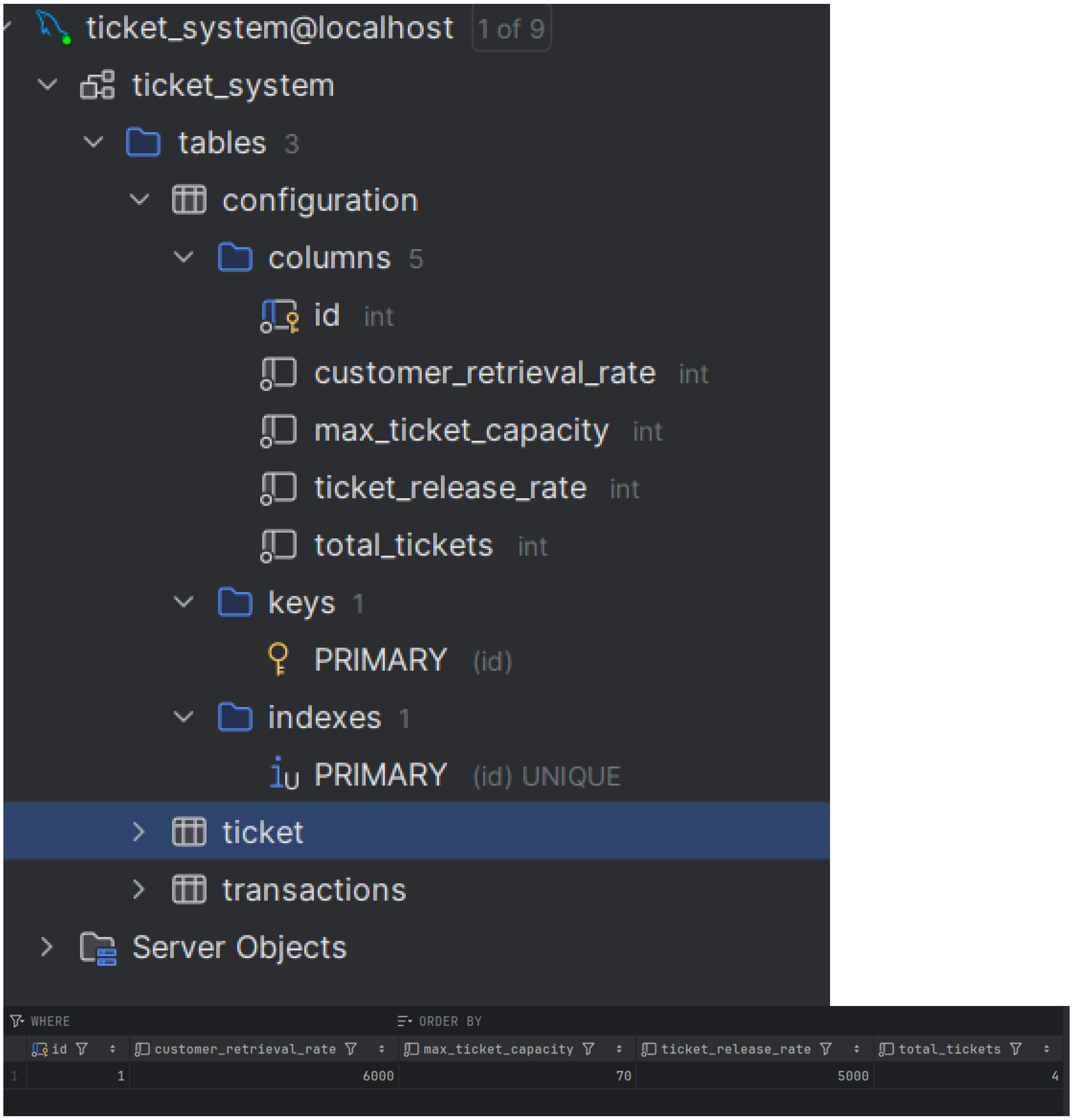
<table>
<tr><td>9</td><td>
Ticket Pool Status
Remaining tickets in the current thread : 2
Maximum Capacity: 70</td></tr>
<tr><td>10</td><td>
Transaction Logs
No transactions available</td></tr>
<tr><td>11</td><td>
Transaction Logs<table>
<tr><th>TRANSACTION ID</th><th>USER</th><th>ACTION</th><th>DATE/TIME</th></tr>
<tr><td>1</td><td>V1</td><td>Added Ticket</td><td>12/8/2024, 5:59:32 PM</td></tr>
<tr><td>2</td><td>V2</td><td>Added Ticket</td><td>12/8/2024, 5:59:32 PM</td></tr>
<tr><td>3</td><td>V1</td><td>Added Ticket</td><td>12/8/2024, 5:59:32 PM</td></tr>
<tr><td>4</td><td>C1</td><td>Purchased Ticket</td><td>12/8/2024, 5:59:32 PM</td></tr>
<tr><td>5</td><td>V2</td><td>Added Ticket</td><td>12/8/2024, 5:59:32 PM</td></tr>
</table></td></tr>
</table>



```
Fetched Transactions:
  ▼ Array(130) [ {…}, {…}, {…},
  {…}, {…}, {…}, {…}, {…}, {…},
  {…}, … ]
    ▶ [0…99]
    ▶ [100…129]
      length: 130
    ▶ <prototype>: Array []

                         App.jsx:45:14
```



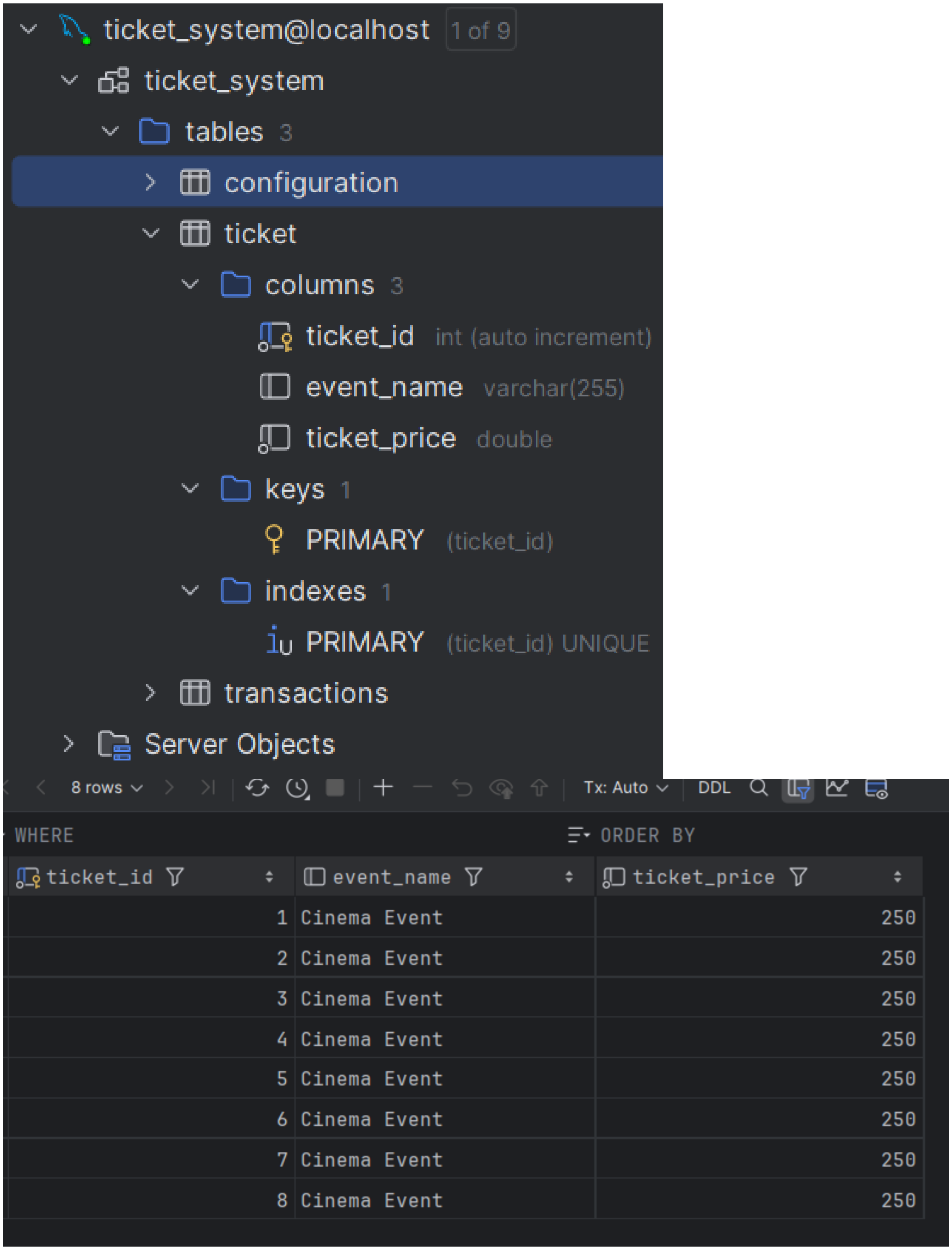
```
Fetched Transactions:
▼ Array(130) [ {…}, {…}, {…},
{…}, {…}, {…}, {…}, {…}, {…},
{…}, … ]
  ▼ [0…99]
    ▶ 0: Object { transactionID:
    1, user: "V1", action: "Added
    Ticket", … }
    ▶ 1: Object { transactionID:
    2, user: "V2", action: "Added
    Ticket", … }
    ▶ 2: Object { transactionID:
    3, user: "V1", action: "Added
    Ticket", … }
    ▶ 3: Object { transactionID:
    4, user: "C1", action:
    "Purchased Ticket", … }
    ▶ 4: Object { transactionID:
    5, user: "V2", action: "Added
    Ticket", … }
    ▶ 5: Object { transactionID:
    6, user: "C1", action:
    "Purchased Ticket", … }
    ▶ 6: Object { transactionID:
    7, user: "V1", action: "Added
```

12	

```
ticket_system@localhost  1 of 9
  ticket_system
    tables 3
      configuration
        columns 5
          id int
          customer_retrieval_rate int
          max_ticket_capacity int
          ticket_release_rate int
          total_tickets int
        keys 1
          PRIMARY (id)
        indexes 1
          PRIMARY (id) UNIQUE
      ticket
      transactions
  Server Objects
```

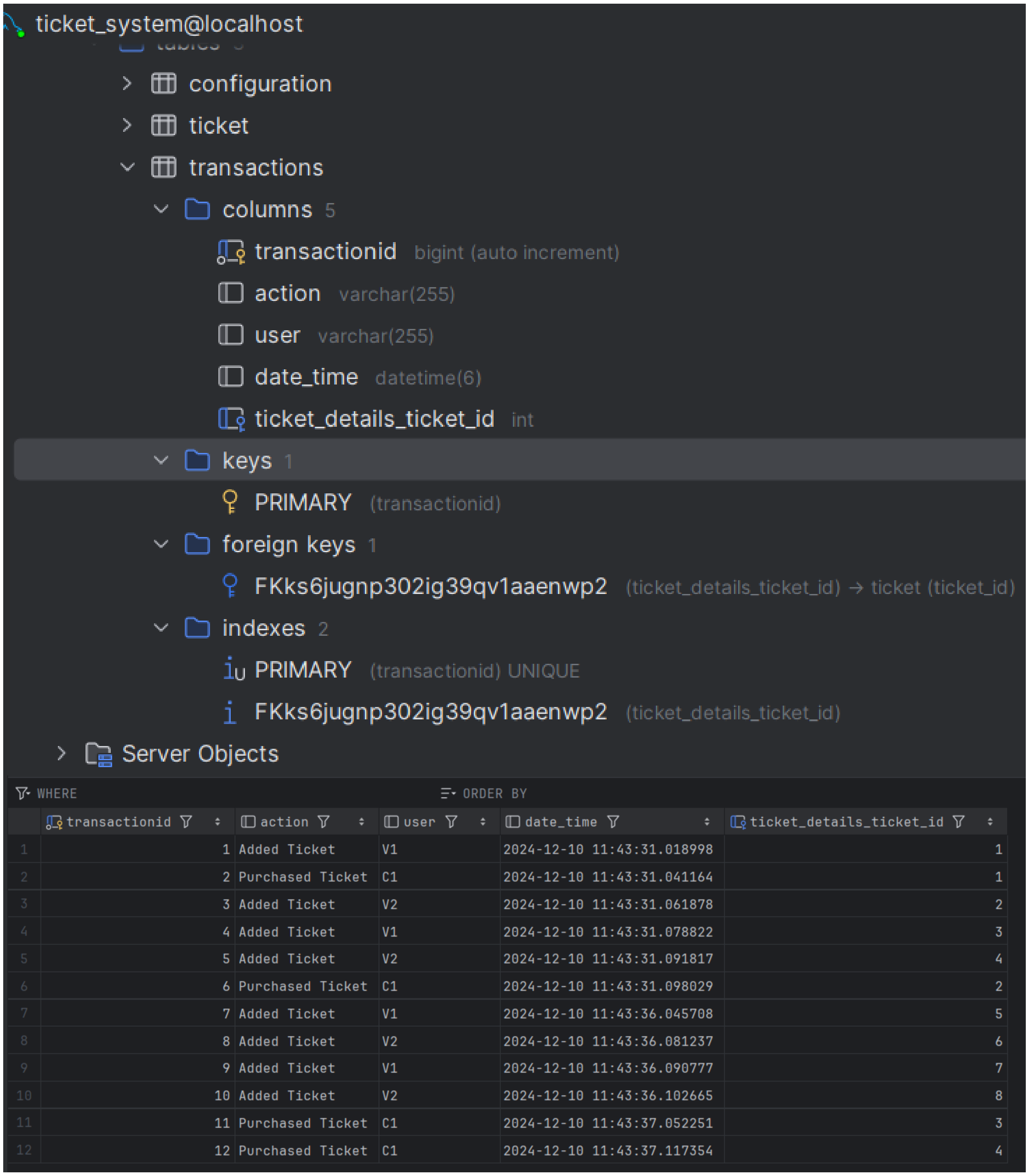
WHERE ORDER BY

	id	customer_retrieval_rate	max_ticket_capacity	ticket_release_rate	total_tickets
1	1	6000	70	5000	4

13	

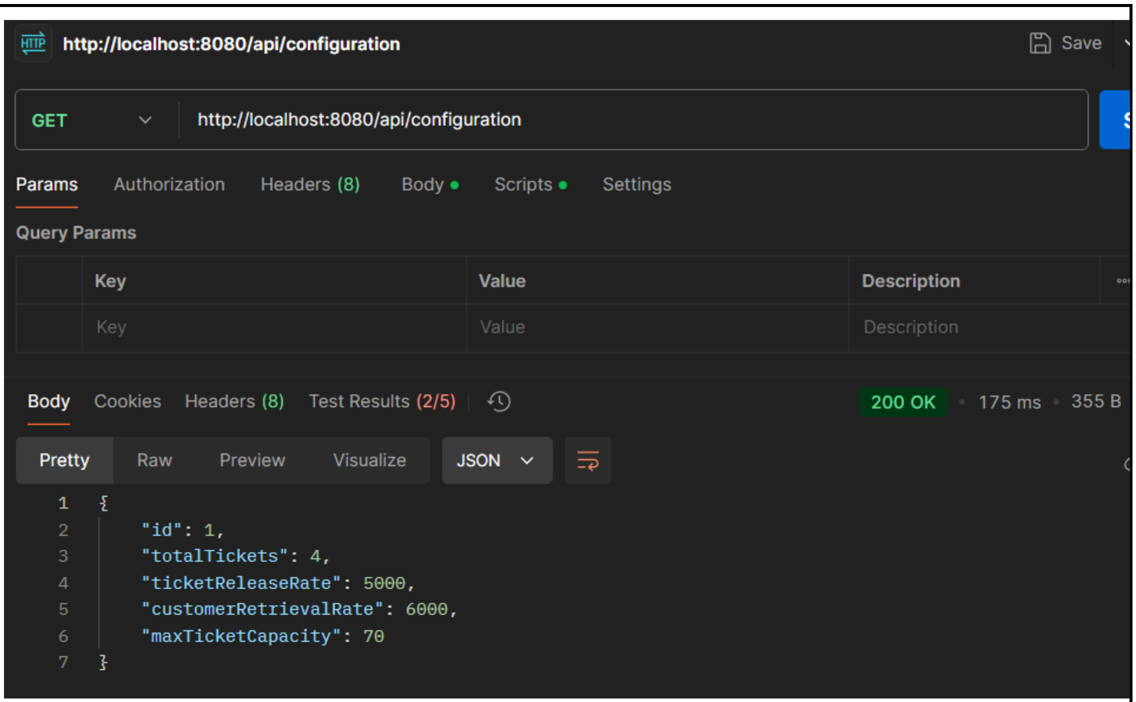
```
ticket_system@localhost  1 of 9
  ticket_system
    tables  3
      configuration
      ticket
        columns  3
          ticket_id  int (auto increment)
          event_name  varchar(255)
          ticket_price  double
        keys  1
          PRIMARY  (ticket_id)
        indexes  1
          PRIMARY  (ticket_id) UNIQUE
      transactions
  Server Objects
```

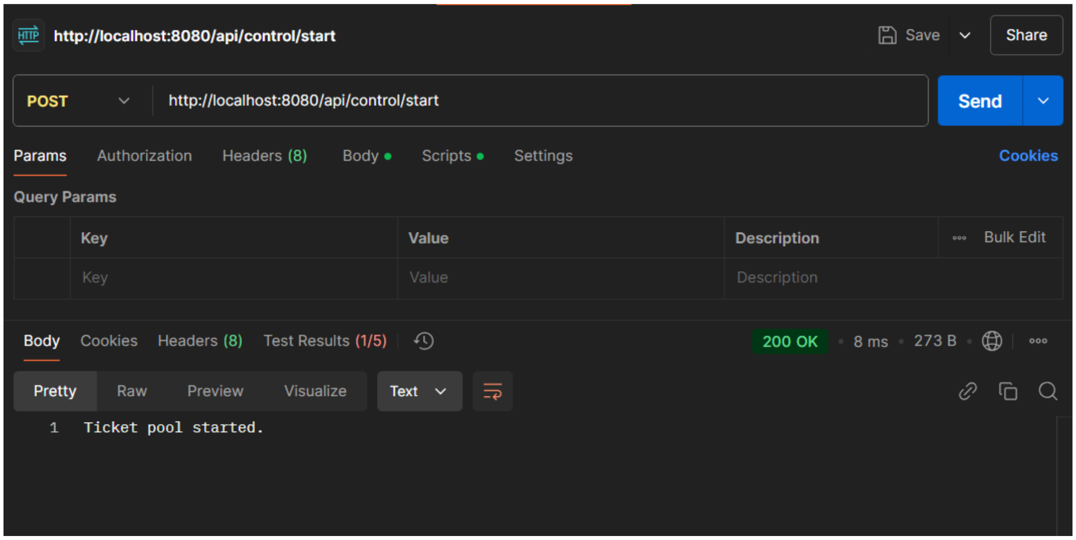
ticket_id	event_name	ticket_price
1	Cinema Event	250
2	Cinema Event	250
3	Cinema Event	250
4	Cinema Event	250
5	Cinema Event	250
6	Cinema Event	250
7	Cinema Event	250
8	Cinema Event	250

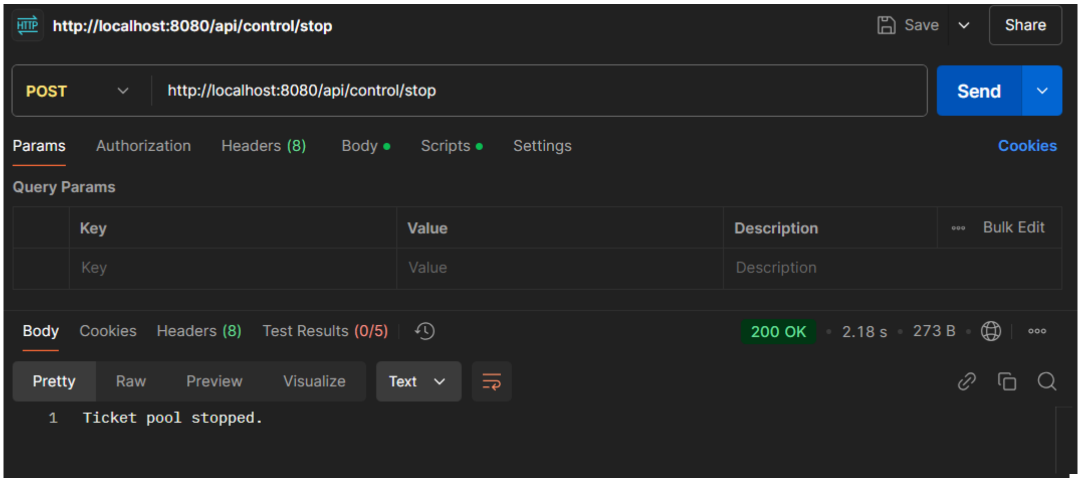
14	

```
ticket_system@localhost
    > configuration
    > ticket
    v transactions
        v columns 5
            transactionid  bigint (auto increment)
            action  varchar(255)
            user  varchar(255)
            date_time  datetime(6)
            ticket_details_ticket_id  int
        v keys 1
            PRIMARY  (transactionid)
        v foreign keys 1
            FKks6jugnp302ig39qv1aaenwp2  (ticket_details_ticket_id) → ticket (ticket_id)
        v indexes 2
            PRIMARY  (transactionid) UNIQUE
            FKks6jugnp302ig39qv1aaenwp2  (ticket_details_ticket_id)
  > Server Objects
```

transactionid	action	user	date_time	ticket_details_ticket_id
1	Added Ticket	V1	2024-12-10 11:43:31.018998	1
2	Purchased Ticket	C1	2024-12-10 11:43:31.041164	1
3	Added Ticket	V2	2024-12-10 11:43:31.061878	2
4	Added Ticket	V1	2024-12-10 11:43:31.078822	3
5	Added Ticket	V2	2024-12-10 11:43:31.091817	4
6	Purchased Ticket	C1	2024-12-10 11:43:31.098029	2
7	Added Ticket	V1	2024-12-10 11:43:36.045708	5
8	Added Ticket	V2	2024-12-10 11:43:36.081237	6
9	Added Ticket	V1	2024-12-10 11:43:36.090777	7
10	Added Ticket	V2	2024-12-10 11:43:36.102665	8
11	Purchased Ticket	C1	2024-12-10 11:43:37.052251	3
12	Purchased Ticket	C1	2024-12-10 11:43:37.117354	4

15	
	http://localhost:8080/api/configuration
	GET http://localhost:8080/api/configuration
	200 OK · 175 ms · 355 B
	{
	    "id": 1,
	    "totalTickets": 4,
	    "ticketReleaseRate": 5000,
	    "customerRetrievalRate": 6000,
	    "maxTicketCapacity": 70
	}

16	 http://localhost:8080/api/control/start POST http://localhost:8080/api/control/start 200 OK · 8 ms · 273 B Ticket pool started.

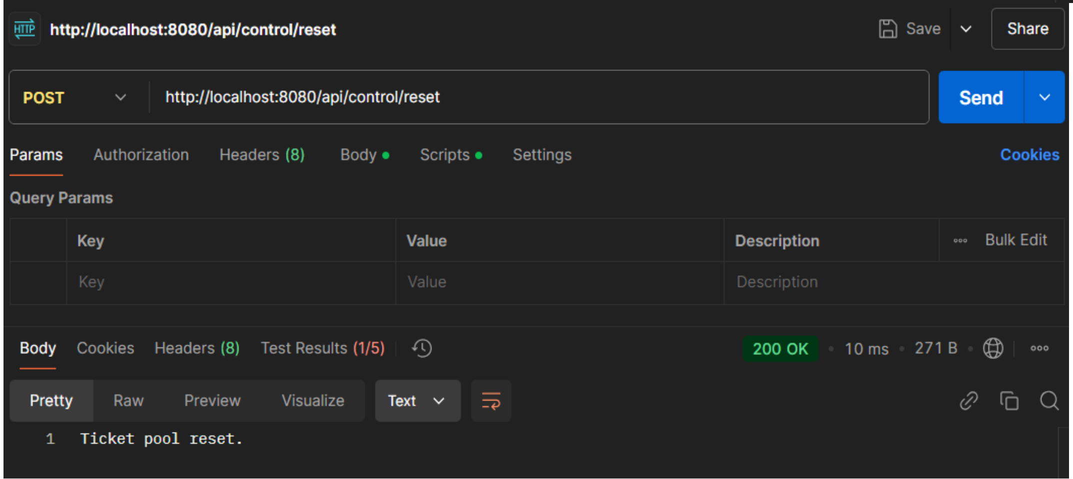


http://localhost:8080/api/control/stop

POST http://localhost:8080/api/control/stop

200 OK • 2.18 s • 273 B

```
Ticket pool stopped.
```

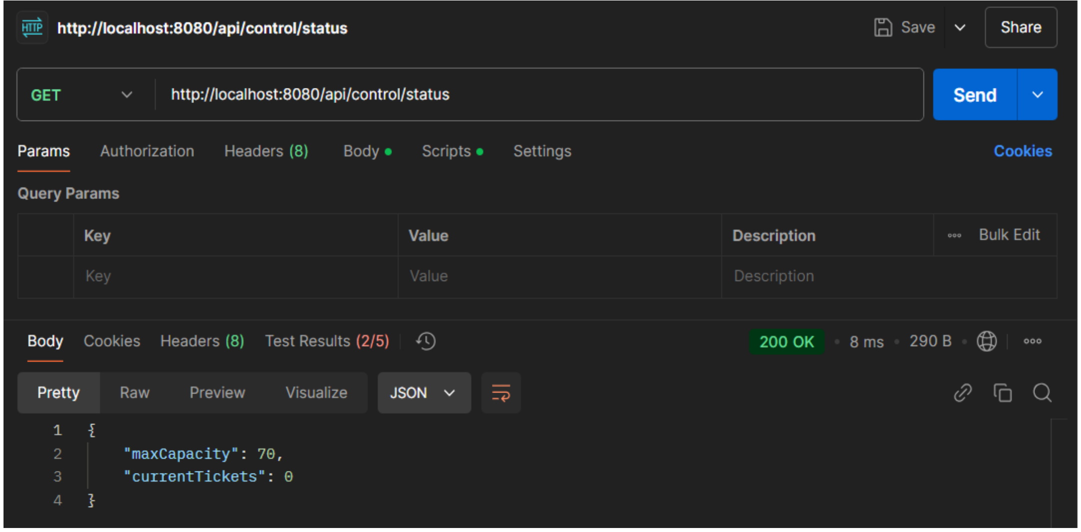


http://localhost:8080/api/control/reset

POST http://localhost:8080/api/control/reset

200 OK • 10 ms • 271 B

```
Ticket pool reset.
```



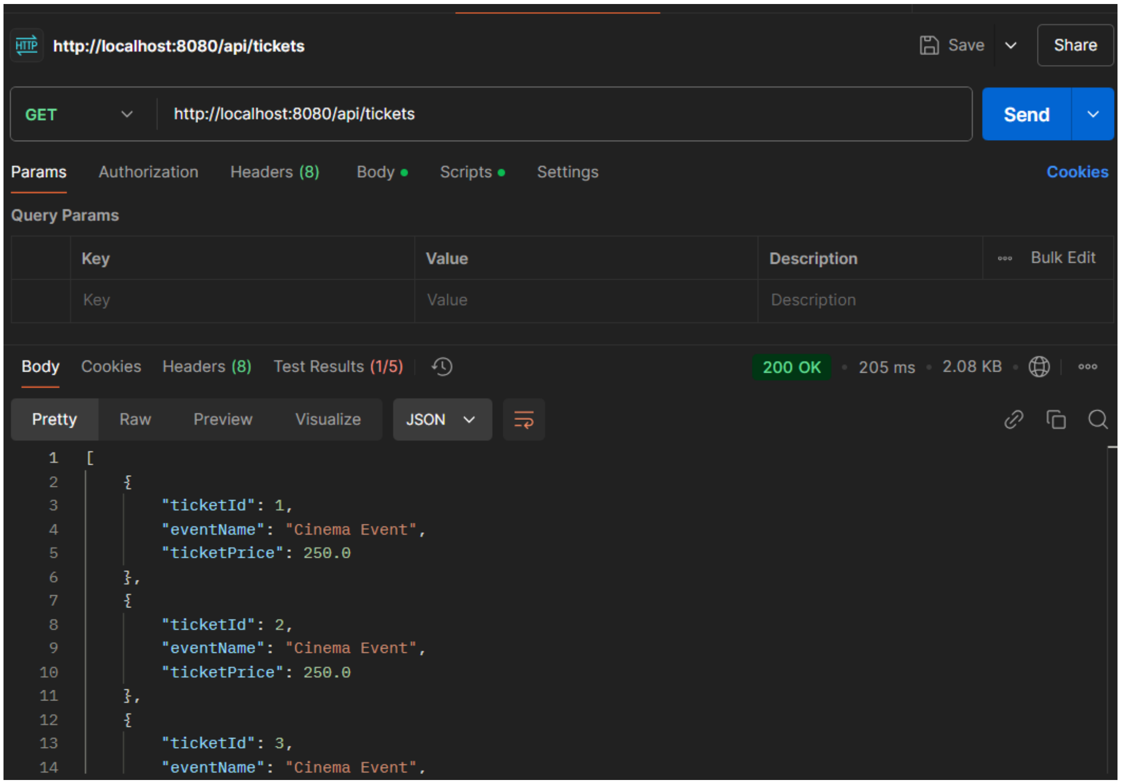
http://localhost:8080/api/control/status

GET http://localhost:8080/api/control/status

200 OK • 8 ms • 290 B

```
{
    "maxCapacity": 70,
    "currentTickets": 0
}
```

17



http://localhost:8080/api/tickets

GET http://localhost:8080/api/tickets

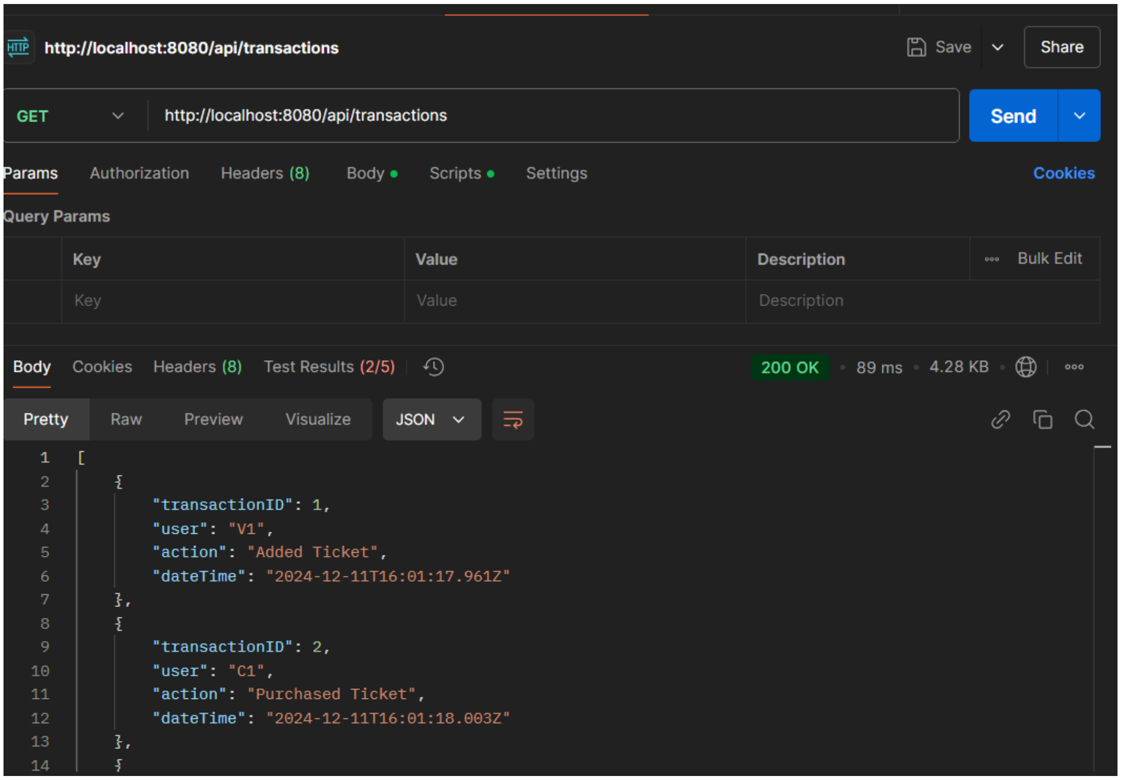
Query Params

Key	Value	Description
Key	Value	Description

200 OK • 205 ms • 2.08 KB

```
[
    {
        "ticketId": 1,
        "eventName": "Cinema Event",
        "ticketPrice": 250.0
    },
    {
        "ticketId": 2,
        "eventName": "Cinema Event",
        "ticketPrice": 250.0
    },
    {
        "ticketId": 3,
        "eventName": "Cinema Event",
```

18



http://localhost:8080/api/transactions

GET http://localhost:8080/api/transactions

Query Params

Key	Value	Description
Key	Value	Description

200 OK • 89 ms • 4.28 KB

```
[
    {
        "transactionID": 1,
        "user": "V1",
        "action": "Added Ticket",
        "dateTime": "2024-12-11T16:01:17.961Z"
    },
    {
        "transactionID": 2,
        "user": "C1",
        "action": "Purchased Ticket",
        "dateTime": "2024-12-11T16:01:18.003Z"
    },
    {
```