



---

## **ADES CA1 Test Case**

**Academic semester: Year 2 Sem 2**

**Module Name:** ST0507 APPLICATION DEVELOPMENT STUDIO

**Module class:** DIT/FT/2B/03

YAP ZONG XIN - P1935574

KEERTIKAA MANOGARANN – P1935714

LIM PEI XIAN - P1936195

**Company: Create Queue**

```
PASS tests/createQueue.test.js
  ✓ success (321 ms)
  ✓ success - company_id minimum (236 ms)
  ✓ success - company_id maximum (239 ms)
  ✓ success - queue_id numeric only (250 ms)
  ✓ success - queue_id alphabets only (240 ms)
  ✓ Queue Id already exists (249 ms)
  ✓ Queue Id already exists - Not case sensitive (948 ms)
  ✓ Error in parameter, company_id, less than 10 digit (6 ms)
  ✓ Error in parameter, company_id, more than 10 digit (7 ms)
  ✓ Error in parameter, company_id, not numeric (5 ms)
  ✓ Error in parameter, company_id, illegal character (4 ms)
  ✓ Error in parameter, queue_id, less than 10 alphanumeric (5 ms)
  ✓ Error in parameter, queue_id, more than 10 alphanumeric (5 ms)
  ✓ Error in parameter, queue_id, illegal characters (4 ms)
  ✓ Error in parameter, queue_id, not string (5 ms)

Test Suites: 1 passed, 1 total
Tests: 15 passed, 15 total
```

**Title:** Create Queue API - Successful request creates queue in database

**Description:** A successful request made to the server should create a new entry in the queues table with the correct parameters.

**Precondition:** Table should be created, queue\_id does not already exist, valid queue\_id (10), valid customer\_id (10).

### Test Steps:

1. Start the Backend Server
2. Send a Create Queue request to the backend with a valid queue\_id
3. Receive a 201 Created response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** A new row with the queue\_id is created

**Example Evidence:**

```
1 HTTP/1.1 201 Created
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: text/html; charset=utf-8
5 Content-Length: 7
6 ETag: W/"7-U6VofLJtxB8qtAM+l+E63v03QNY"
7 Date: Fri, 27 Nov 2020 02:53:13 GMT
8 Connection: close
9
10 success
```

SELECT \* FROM "public"."queues" LIMIT 100

Table queries ▾

Previous queries ▾

Execute ▶

id	queue_id	status	company_id
1	QUEUE12345	INACTIVE	1234567890

**Title:** Create Queue API – Unsuccessful request due to duplicate queue\_id

**Description:** An unsuccessful request made to the server should display an error message “Queue Id: (queue\_id) already exists”.

**Precondition:** Table should be created, queue\_id exists, valid queue\_id (10), valid customer\_id (10).

**Test Steps:**

1. Start the Backend Server
2. Send a Create Queue request to the backend with an existing queue\_id
3. Receive a 422 Unprocessable Entity response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** No new row with the queue\_id is created

**Example Evidence:**

```
1 HTTP/1.1 422 Unprocessable Entity
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-b5dYm4rhxbtQzsKp8SCgilKmyZE"
7 Date: Fri, 27 Nov 2020 02:59:45 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12345 already exists",
12   "code": "QUEUE_EXISTS"
13 }
```

(no new rows created)

**Title:** Create Queue API – Unsuccessful request due to invalid queue\_id

**Description:** An unsuccessful request made to the server should display an error message “You have an invalid JSON body!”

**Precondition:** Table should be created, customer\_id is valid (10), queue\_id is invalid (less/more than 10).

**Test Steps:**

1. Start the Backend Server
2. Send a Create Queue request to the backend with an invalid queue\_id
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the queue tables
5. **Expected Result:** No new row with the queue\_id is created

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQaO2qYI/bRYaQl9gxONKjo"
7 Date: Fri, 27 Nov 2020 03:02:24 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no new rows is created)

**Title:** Create Queue API – Unsuccessful request due to invalid customer\_id

**Description:** An unsuccessful request made to the server should display an error message “You have an invalid JSON body!”

**Precondition:** Table should be created, queue\_id is valid (10), customer\_id is invalid (less/more than 10 digits).

**Test Steps:**

1. Start the Backend Server
2. Send a Create Queue request to the backend with an invalid queue\_id
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the queue tables
5. **Expected Result:** No new row with the queue\_id is created

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQa02qYI/bRYaQ19gxONKjo"
7 Date: Fri, 27 Nov 2020 03:04:24 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no new rows created)

## Company: Update Queue

```
PASS tests/updateQueue.test.js (5.654 s)
  ✓ success (271 ms)
  ✓ success - queue_id number only (276 ms)
  ✓ success - queue_id alphabet only (267 ms)
  ✓ success - activated remains activated (274 ms)
  ✓ success - deactivate (265 ms)
  ✓ success - deactivated remains deactivated (267 ms)
  ✓ success - not case sensitive (265 ms)
  ✓ success - deactivated remains deactivated (262 ms)
  ✓ Error - queue id not found (260 ms)
  ✓ Error - queue id less than 10 digits (5 ms)
  ✓ Error - queue id more than 10 digits (5 ms)
  ✓ Error - queue id not alpha numeric (4 ms)
  ✓ Error - status not DEACTIVATE or ACTIVATE (5 ms)
  ✓ Error - status not DEACTIVATE or ACTIVATE (5 ms)

Test Suites: 1 passed, 1 total
Tests: 14 passed, 14 total
```

**Title:** Update Queue API - Successful request updates the queue status in database (TO ACTIVE)

**Description:** A successful request made to the server should update an entry in the queues table with the status as 'ACTIVATED'

**Precondition:** Table should be created, queue\_id is valid (10), queue\_id exists, status is inactive, status is valid.

### Test Steps:

1. Start the Backend Server
2. Send an update Queue request to the backend with status as "ACTIVATE"
3. Receive a 200 OK response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** A row with the queue\_id has its status updated to "ACTIVE"

### Example Evidence:

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 43
6 ETag: W/"2b-6CCSmEYvU6Qi3isx/dWnfri4KUU"
7 Date: Fri, 27 Nov 2020 03:08:10 GMT
8 Connection: close
9
10 {
11   "queue_id": "QUEUE12345",
12   "status": "ACTIVE"
13 }
```

(successfully updated to active)

SELECT * FROM "public"."queues" LIMIT 100			
Table queries ▾ Previous queries ▾			
Execute ▶			
id	queue_id	status	company_id
1	QUEUE12345	ACTIVE	1234567890

**Title:** Update Queue API - Successful request updates the queue status in database (TO INACTIVE)

**Description:** A successful request made to the server should update an entry in the queue\_tab with the status

**Precondition:** Table should be created, queue\_id is valid (10), queue\_id exists, status is active, status is valid.

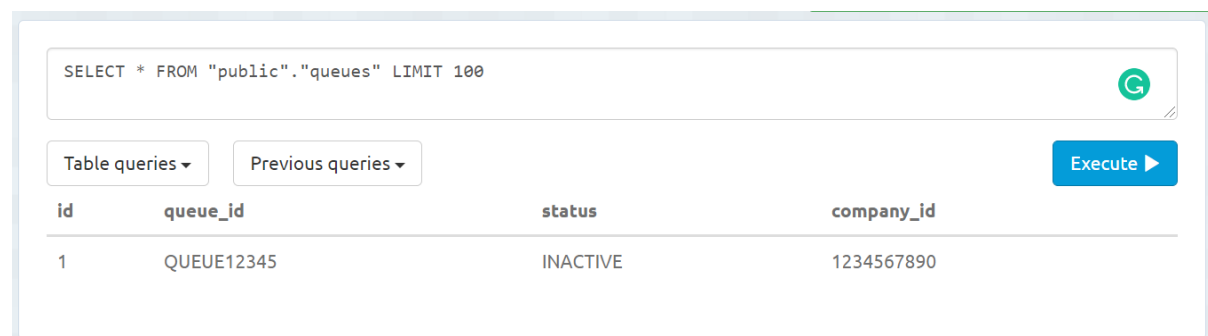
**Test Steps:**

1. Start the Backend Server
2. Send an update Queue request to the backend with status as "DEACTIVATE"
3. Receive a 200 OK response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** A row with the queue\_id has its status updated to "INACTIVE"

**Example Evidence:**

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 45
6 ETag: W/"2d-uzS4tuBGCVuSTNgQRQOh//z5px0"
7 Date: Fri, 27 Nov 2020 03:27:58 GMT
8 Connection: close
9
10 {
11   "queue_id": "QUEUE12345",
12   "status": "INACTIVE"
13 }
```

(successfully updated to inactive)



The screenshot shows the Elephantsql web interface. At the top, a SQL query is entered: `SELECT * FROM "public"."queues" LIMIT 100`. Below the query input, there are buttons for "Table queries" and "Previous queries", and an "Execute" button. The results are displayed in a table with the following columns: `id`, `queue_id`, `status`, and `company_id`. The table contains one row with the following values: `1`, `QUEUE12345`, `INACTIVE`, and `1234567890`.

id	queue_id	status	company_id
1	QUEUE12345	INACTIVE	1234567890

**Title:** Update Queue API - Unsuccessful request due to invalid status

**Description:** An unsuccessful request made to the server should display an error message “You have an invalid JSON string!”

**Precondition:** Table should be created, queue\_id is valid (10), queue\_id already exists, status is inactive, status is invalid.

**Test Steps:**

1. Start the Backend Server
2. Send an update Queue request to the backend with status as “ACTIVE”
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** The status of a row with the queue\_id has not been updated to “ACTIVE”

**Example Evidence:**

```
1  HTTP/1.1 400 Bad Request
2  X-Powered-By: Express
3  Access-Control-Allow-Origin: *
4  Content-Type: application/json; charset=utf-8
5  Content-Length: 69
6  ETag: W/"45-mrKYzQaO2qYI/bRYaQl9gxONKjo"
7  Date: Fri, 27 Nov 2020 03:11:07 GMT
8  Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no changes to the row)



**Title:** Update Queue API - Unsuccessful request due to invalid queue\_id

**Description:** An unsuccessful request made to the server should display an error message “You have an invalid JSON string!”

**Precondition:** Table should be created, queue\_id already exists, status is inactive, status is valid, queue\_id is invalid (less/more than 10).

**Test Steps:**

1. Start the Backend Server
2. Send an update Queue request to the backend with status as “ACTIVE”
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** The status of a row with the queue\_id has not been updated to “ACTIVE”

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQaO2qYI/bRYaQl9gxONKjo"
7 Date: Fri, 27 Nov 2020 03:13:33 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no changes to the row)

**Title:** Update Queue API - Unsuccessful request due to unknown queue\_id

**Description:** An unsuccessful request made to the server should display an error message "Queue Id: (queue\_id) Not Found"

**Precondition:** Table should be created, status is inactive, status is valid, queue\_id is valid.

**Test Steps:**

1. Start the Backend Server
2. Send an update Queue request to the backend with status as "ACTIVE"
3. Receive a 404 Not Found response
4. Go to elephantsql to inspect the queues table
5. **Expected Result:** The status of a row with the queue\_id has not been updated to "ACTIVE"

**Example Evidence:**

```
1 HTTP/1.1 404 Not Found
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-qhDP19/vkTdJInLN3AvWUNk1nYE"
7 Date: Fri, 27 Nov 2020 03:23:18 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12346 Not Found",
12   "code": "UNKNOWN_QUEUE"
13 }
```

(no changes to the row)

## Customer: Join Queue

```
PASS tests/joinQueue.test.js (10.848 s)
  ✓ success (492 ms)
  ✓ success - multiple people joining queue (1882 ms)
  ✓ success - customer_id minimum (503 ms)
  ✓ success - customer_id maximum (486 ms)
  ✓ success - queue_id numeric only (491 ms)
  ✓ success - queue_id alphabets only (483 ms)
  ✓ Error - Queue Id dont exists (255 ms)
  ✓ Error - Queue not activated (239 ms)
  ✓ Error in parameter, customer_id, less than 10 digit (17 ms)
  ✓ Error in parameter, customer_id, more than 10 digit (6 ms)
  ✓ Error in parameter, customer_id, not numeric (7 ms)
  ✓ Error in parameter, customer_id, illegal character (4 ms)
  ✓ Error in parameter, queue_id, less than 10 alphanumeric (4 ms)
  ✓ Error in parameter, queue_id, more than 10 alphanumeric (6 ms)
  ✓ Error in parameter, queue_id, illegal characters (4 ms)
  ✓ Error in parameter, queue_id, not a string (4 ms)

Test Suites: 1 passed, 1 total
Tests: 16 passed, 16 total
```

**Title:** Join Queue API – A successful request allows customer to join a queue.

**Description:** A successful request made to the server should create a new entry in customers tab with correct parameters.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid, queue status is active.

### Test Steps:

1. Start the Backend Server
2. Send joinQueue request to the backend with a valid queue\_id and customer\_id.
3. Receive a 201 Created response
4. Go to elephantsql to inspect the customers tab
5. **Expected Result:** A new row with queue\_id and customer\_id is created.

### Example Evidence:

```
1 HTTP/1.1 201 Created
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 50
6 ETag: W/"32-7KXvLsVvRcQzhGZtjsekizZAcj4"
7 Date: Fri, 27 Nov 2020 03:57:52 GMT
8 Connection: close
9
10 {
11   "customer_id": 1234567890,
12   "queue_id": "QUEUE12345"
13 }
```

(new row is created)

SELECT * FROM "public"."customers" LIMIT 100				
Table queries ▾		Previous queries ▾		Execute ▶
id	customer_id	queue_id	jointime	serveravailable
1	1234567890	QUEUE12345	2020-11-27 03:57:50 +0000	false

**Title:** Join Queue API – An unsuccessful request that does not allow customer to join a non-existent queue.

**Description:** An unsuccessful request made to the sever should not allow customer to join a non-existent queue. It should not create a new entry in the customers tab. Error message together with 404 Not Found should be displayed.

**Precondition:** Table should be created, queue\_id is valid, customer\_id is valid, queue status is active.

**Test Steps:**

1. Start the Backend Server
2. Send joinQueue request with a customer\_id and a non-existent queue\_id
3. Receive a 404 Not Found response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** No new row is created

**Example Evidence:**

```
1 HTTP/1.1 404 Not Found
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-qhDP19/vkTdJInLN3AvWUNk1nYE"
7 Date: Fri, 27 Nov 2020 04:06:29 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12346 Not Found",
12   "code": "UNKNOWN_QUEUE"
13 }
```

(no rows are created)

**Title:** Join Queue API – An unsuccessful request that does not allow customer to join queue they are already in.

**Description:** An unsuccessful request made to the sever should not allow customer to join a queue that they are already in. It should not create a new entry in the customers tab.

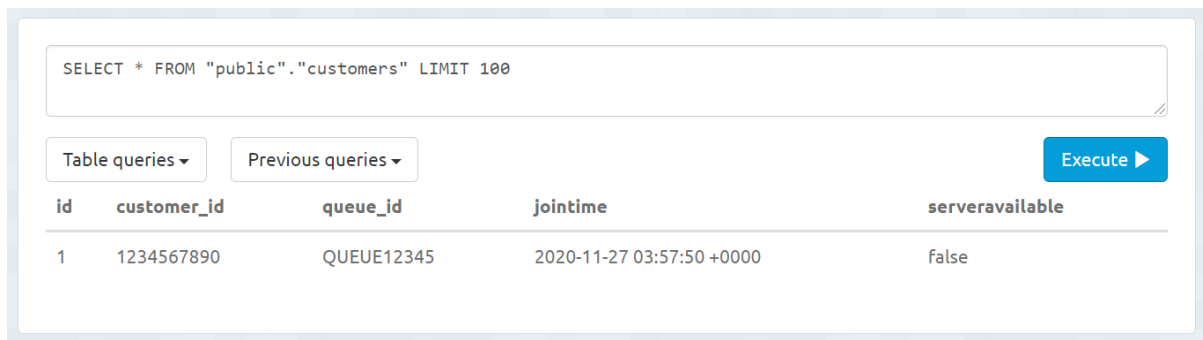
**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid, queue status is active customer already in queue.

**Test Steps:**

1. Start the Backend Server
2. Send joinQueue request to a row of queue\_id which the customer\_id is already in
3. Receive a 422 Unprocessable Entity Failed response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** No new row is created

**Example Evidence:**

(no rows are created)



The screenshot shows the ElephantSQL interface. At the top, a text box contains the SQL query: `SELECT * FROM "public"."customers" LIMIT 100`. Below the text box are two dropdown menus labeled "Table queries" and "Previous queries", and a blue "Execute" button with a right-pointing arrow. Below these elements is a table with the following columns: `id`, `customer_id`, `queue_id`, `jointime`, and `serveravailable`. The table contains one row of data.

<code>id</code>	<code>customer_id</code>	<code>queue_id</code>	<code>jointime</code>	<code>serveravailable</code>
1	1234567890	QUEUE12345	2020-11-27 03:57:50 +0000	false

**Title:** Join Queue API – An unsuccessful request that does not allow customer to join an inactive queue.

**Description:** An unsuccessful request made to the sever should not allow customer to join a queue that is inactive. It should not create a new entry in the customers tab. Error message together with 422 Unprocessable Entity should be displayed.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid, queue status is active, customer already in queue.

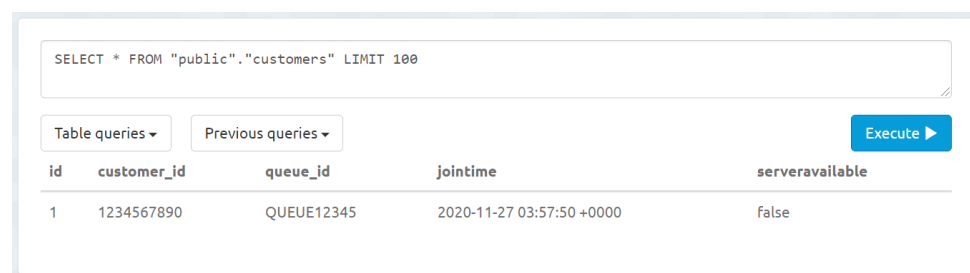
**Test Steps:**

1. Start the Backend Server
2. Send joinQueue request to a row of queue\_id which the customer\_id is already in
3. Receive a 422 Unprocessable Entity response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** No new row is created

**Example Evidence:**

```
1 HTTP/1.1 422 Unprocessable Entity
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-fYUyufADFjFV/JYOL45YiB+/etg"
7 Date: Fri, 27 Nov 2020 04:21:38 GMT
8 Connection: close
9
10 {
11   "error": "Queue QUEUE12345 Is Inactive!",
12   "code": "INACTIVE_QUEUE"
13 }
```

(no rows are created)



The screenshot shows the Elephantsql web interface. At the top, a text box contains the SQL query: `SELECT * FROM "public"."customers" LIMIT 100`. Below the text box are two buttons: "Table queries" and "Previous queries". To the right of these buttons is a blue "Execute" button with a play icon. Below the buttons is a table with the following columns: `id`, `customer_id`, `queue_id`, `jointime`, and `serveravailable`. The table contains one row of data.

id	customer_id	queue_id	jointime	serveravailable
1	1234567890	QUEUE12345	2020-11-27 03:57:50 +0000	false

**Title:** Join Queue API – An unsuccessful request does not allow customer to join a queue with invalid parameter error in queue\_id.

**Description:** An unsuccessful request made to the server should not allow customer to join a queue when the queue\_id is invalid. It should not create a new entry in the customers tab. Error message together with 400 Bad Request should be displayed. (e.g. queue\_id less than 10)

**Precondition:** Table should be created, queue\_id already exists, customer\_id is valid, queue status is active.

**Test Steps:**

1. Start the Backend Server
2. Send joinQueue request with a valid customer\_id but an invalid queue\_id
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** No new row is created

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQa02qYI/bRYaQl9gxONKjo"
7 Date: Fri, 27 Nov 2020 04:30:12 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no rows are created)

**Title:** Join Queue API – An unsuccessful request does not allow customer to join a queue with invalid parameter error in customer\_id.

**Description:** An unsuccessful request made to the server should not allow customer to join a queue when the customer\_id is invalid. It should not create a new entry in the customers tab. Error message together with 400 Bad Request should be displayed. (e.g. customer\_id less than 10)

**Precondition:** Table should be created, queue\_id already exists, queue\_id is valid, queue status is active.

**Test Steps:**

1. Start the Backend Server
2. Send joinQueue request with a valid queue\_id but an invalid customer\_id
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** No new row is created

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQaO2qYI/bRYaQl9gxONKjo"
7 Date: Fri, 27 Nov 2020 04:38:27 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(no rows are created)



## Company: Server Available

```
PASS tests/serverAvailable.test.js (9.357 s)
  ✓ success (713 ms)
  ✓ success - 2nd poll (701 ms)
  ✓ success - queue_id all number (473 ms)
  ✓ success - queue_id all character (485 ms)
  ✓ success - queue_id not case sensitive (710 ms)
  ✓ Error - queue_id does not exists (246 ms)
  ✓ Error - queue_id too short (11 ms)
  ✓ Error - queue_id too long (4 ms)
  ✓ Error - queue_id not string (6 ms)
Test Suites: 1 passed, 1 total
Tests: 9 passed, 9 total
```

**Title:** Server Available API - Successful request updates the server available status in database so that it allows the next customer into the server (remove first row customer in the queue)

**Description:** A successful request updates the server available status in database so that it allows the next customer into the server (remove first row customer in the queue). Hence it updates the entry in the customer tab by removing the first customer in the queue.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists queue\_id is valid.

### Test Steps:

1. Start the Backend Server
2. Send a server available request to the backend with a queue\_id that is available
3. Receive a 200 OK response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** The first row of the customer table has been removed/went into the server.

### Example Evidence:

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 26
6 ETag: W/"1a-7rl0SXRob2dVKZLonxGq6TowFS8"
7 Date: Fri, 27 Nov 2020 04:52:03 GMT
8 Connection: close
9
10 {
11   "customer_id": 1234567890
12 }
```

(first row in the table is removed from the queue/went into the available server)

Table queries ▾

Previous queries ▾

Execute ▶

No rows returned

**Title:** Server Available API – An unsuccessful request does not allow customer to enter a server that is available due to invalid queue\_id.

**Description:** An unsuccessful request does not allow customer to enter a server that is available due to invalid parameter error in queue\_id. It should not remove the first row in the customers tab. Error message together with 400 Bad Request should be displayed. (e.g. queue\_id less than 10)

**Precondition:** Table should be created, queue\_id is invalid.

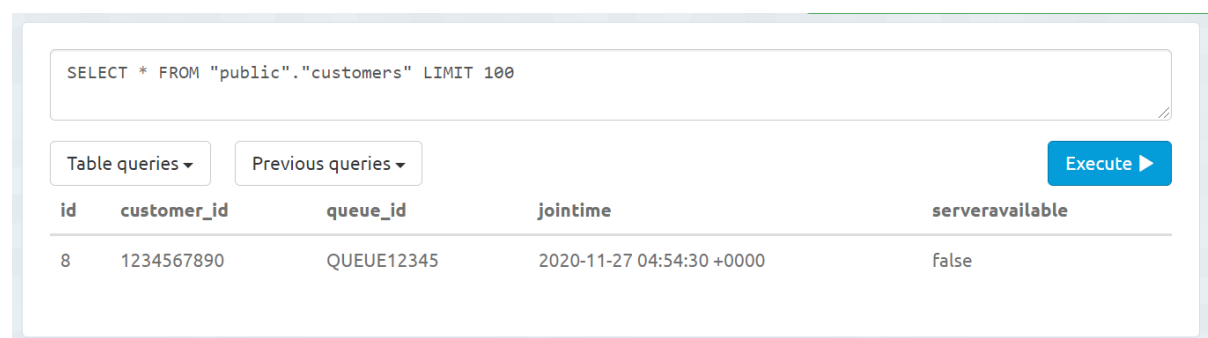
**Test Steps:**

1. Start the Backend Server
2. Send a server available request to the backend with an invalid queue\_id
3. Receive a 400 Bad Request response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** The first row of the customer table has not been removed/has not went into the server.

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 69
6 ETag: W/"45-mrKYzQa02qYI/bRYaQl9gxONKjo"
7 Date: Fri, 27 Nov 2020 04:54:45 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid JSON body!",
12   "code": "INVALID_JSON_BODY"
13 }
```

(customer was not removed)



The screenshot shows the ElephantSQL interface. At the top, a text box contains the SQL query: `SELECT * FROM "public"."customers" LIMIT 100`. Below the text box are two dropdown menus labeled "Table queries" and "Previous queries", and a blue "Execute" button with a right-pointing triangle. Below these elements is a table with the following data:

id	customer_id	queue_id	jointime	serveravailable
8	1234567890	QUEUE12345	2020-11-27 04:54:30 +0000	false

**Title:** Server Available API – An unsuccessful request does not allow customer to enter a server that is available due to unknown queue\_id.

**Description:** An unsuccessful request does not allow customer to enter a server that is available due to unknown parameter error in queue\_id. It should not remove the first row in the customers tab. Error message together with 404 Not Found should be displayed.

**Precondition:** Table should be created, queue\_id is unknown, queue\_id is valid.

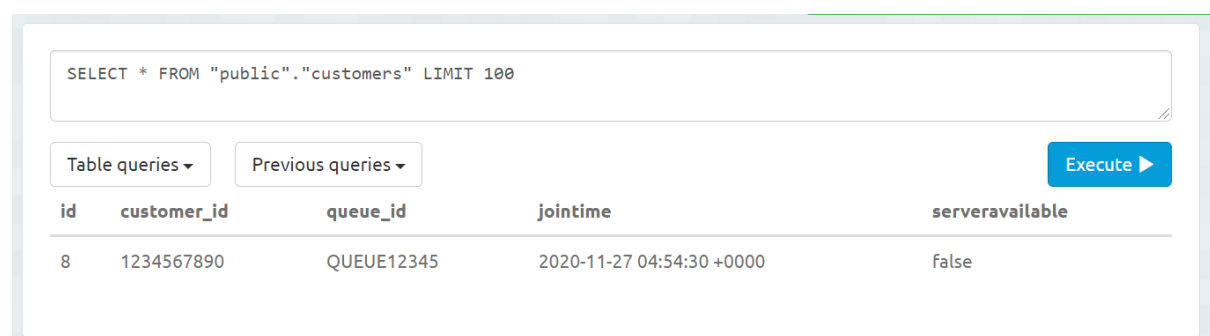
**Test Steps:**

1. Start the Backend Server
2. Send a server available request to the backend with an unknown queue\_id
3. Receive a 404 Not Found response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** The first row of the customer table has not been removed/has not went into the server.

**Example Evidence:**

```
1 HTTP/1.1 404 Not Found
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-qhDP19/vkTdJJInLN3AvWUNk1nYE"
7 Date: Fri, 27 Nov 2020 04:59:40 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12346 Not Found",
12   "code": "UNKNOWN_QUEUE"
13 }
```

(customer was not removed)



The screenshot shows the ElephantSQL interface. At the top, a SQL query is entered in a text box: `SELECT * FROM "public"."customers" LIMIT 100`. Below the query box are two buttons: "Table queries" and "Previous queries". To the right of these buttons is a blue "Execute" button with a play icon. Below the buttons is a table displaying the results of the query. The table has five columns: "id", "customer\_id", "queue\_id", "jointime", and "serveravailable". The first row of data shows an id of 8, a customer\_id of 1234567890, a queue\_id of QUEUE12345, a jointime of 2020-11-27 04:54:30 +0000, and a serveravailable status of false.

id	customer_id	queue_id	jointime	serveravailable
8	1234567890	QUEUE12345	2020-11-27 04:54:30 +0000	false

## Customer: Check Queue

```
PASS tests/checkQueue.test.js (9.874 s)
✓ success (303 ms)
✓ success (329 ms)
✓ success - queueid not case sensitive (305 ms)
✓ success - customer minimum (304 ms)
✓ success - customer maximum (301 ms)
✓ success - customer maximum (301 ms)
✓ success - queue_id all number (308 ms)
✓ success - queue_id all alphabet (305 ms)
✓ error - queue_id not found (306 ms)
✓ error - customer_id too small (7 ms)
✓ error - customer_id too big (5 ms)
✓ error - customer_id contain alphabets (4 ms)
✓ error - queue_id too short (4 ms)
✓ error - queue_id too long (5 ms)
✓ error - queue_id invalid character (7 ms)
```

```
Test Suites: 1 passed, 1 total
Tests: 15 passed, 15 total
```

**Title:** Check Queue API – A successful request allow customer to check the number of people queue. (BOTH CUSTOMER\_ID AND QUEUE\_ID ENTERED)

**Description:** A successful request made should display the total number of people in a queue, the number of people ahead of a customer and the status of a queue when a valid queue\_id and customer\_id are entered.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid, customer\_id already exists.

### Test Steps:

1. Start the Backend Server
2. Send checkQueue request to the backend with a valid queue\_id and customer\_id.
3. Receive a 200 OK response
4. Go to elephantsql to inspect the customers table
5. **Expected Result:** It should reflect the correct queue order information of the customer\_id and queue\_id.

### Example Evidence:

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 39
6 ETag: W/"27-Lo2TiEQlytbs7YLWhXkq34d10M8"
7 Date: Fri, 27 Nov 2020 05:47:30 GMT
8 Connection: close
9
10 {
11   "total": 3,
12   "ahead": 2,
13   "status": "ACTIVE"
14 }
```

SELECT * FROM "public"."customers" LIMIT 100				
Table queries ▾		Previous queries ▾		Execute ▶
id	customer_id	queue_id	jointime	serveravailable
1	1234567890	QUEUE12345	2020-11-27 05:44:53 +0000	false
2	1234567899	QUEUE12345	2020-11-27 05:47:08 +0000	false
4	1234567891	QUEUE12345	2020-11-27 05:47:20 +0000	false

**Title:** Check Queue API – A successful request allow customer to check the number of people queue.  
(ONLY QUEUE\_ID ENTERED)

**Description:** A successful request made to the server should display the total number of people in a queue, the number of people ahead and the status of a queue when a valid queue\_id is entered without providing customer\_id.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists.

**Test Steps:**

1. Start the Backend Server
2. Send checkQueue request to the backend with a valid queue\_id.
3. Receive a 200 OK response
4. **Expected Result:** 200 ok success response is displayed together with the total number of people in the queue, the number of people ahead of the customer as '-1' and the status of the queue.

**Example Evidence:**

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 40
6 ETag: W/"28-S359NOowhWkwnwjQbVpMpsIwgic"
7 Date: Fri, 27 Nov 2020 06:04:55 GMT
8 Connection: close
9
10 {
11   "total": 3,
12   "ahead": -1,
13   "status": "ACTIVE"
14 }
```

**Title:** Check Queue API – A successful request to allow customer to know the number of people in a queue when customer inquiring is next to be assigned a server. (BOTH CUSTOMER\_ID AND QUEUE\_ID ENTERED)

**Description:** A successful request made to the server should display the total number of people in a queue, the number of people ahead of a customer and the status of a queue when a valid queue\_id is entered and when customer with customer\_id entered is next to be assigned a server.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid, customer\_id already exists, customer\_id is at the first row.

**Test Steps:**

1. Start the Backend Server
2. Send checkQueue get request with a valid queue\_id and customer\_id which is next to be assigned a server.
3. Receive a 200 OK response
4. **Expected Result:** 200 ok success response is displayed together with the total number of people in the queue, the number of people ahead of the customer as '0' and the status of the queue.

**Example Evidence:**

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 39
6 ETag: W/"27-s5QXgJC11fAI9lNy4LQ8K3aHjKw"
7 Date: Fri, 27 Nov 2020 06:13:15 GMT
8 Connection: close
9
10 {
11   "total": 3,
12   "ahead": 0,
13   "status": "ACTIVE"
14 }
```

**Title:** Check Queue API – A successful request to allow customer to know the number of people in a queue when customer inquiring has never joined or missed a queue.

**Description:** A successful request made to the server should display the total number of people in a queue, the number of people ahead of a customer and the status of a queue when a valid queue\_id is entered and customer\_id that was not in or has missed the queue.

**Precondition:** Table should be created, queue\_id is valid, queue\_id already exists, customer\_id is valid.

**Test Steps:**

1. Start the Backend Server
2. Send checkQueue get request with a valid queue\_id and customer\_id that was not in or has missed the queue.
3. Receive a 200 OK response
4. **Expected Result:** 200 ok success response is displayed together with the total number of people in the queue, the number of people ahead of the customer as '-1' and the status of the queue.

**Example Evidence:**

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 40
6 ETag: W/"28-S359NOowhWkwnwjQbVpMpsIwgic"
7 Date: Fri, 27 Nov 2020 06:17:25 GMT
8 Connection: close
9
10 {
11   "total": 3,
12   "ahead": -1,
13   "status": "ACTIVE"
14 }
```

**Title:** Check Queue API – An unsuccessful request does not allow customer to know the number of people in a queue when there is an invalid parameter error in queue\_id.

**Description:** An unsuccessful request does not allow customer to know the number of people in a queue when an invalid queue\_id is entered. Error message together with 400 Bad Request should be displayed.

**Precondition:** Table should be created, queue\_id is invalid, customer\_id is valid.

**Test steps:**

1. Start Backend Server.
2. Send checkQueue get request with an invalid queue\_id.
3. Receive a 400 Bad request response.
4. **Expected Result:** 400 Bad Request Failed response is displayed together with the error message: 'ID should be 10-digits!'.

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 64
6 ETag: W/"40-11Tl6ZvHpwDnmc5N6+CE0jH9nyw"
7 Date: Fri, 27 Nov 2020 06:21:03 GMT
8 Connection: close
9
10 {
11   "error": "ID should be 10-digits",
12   "code": "INVALID_QUERY_STRING"
13 }
```



**Title:** Check Queue API – An unsuccessful request does not allow customer to know the number of people in a queue when there is a parameter error in customer\_id.

**Description:** An unsuccessful request does not allow customer to know the number of people in a queue when a valid queue\_id is entered when an invalid customer\_id is provided. Error message together with 400 Bad Request should be displayed.

**Precondition:** Valid queue\_id should be entered but an invalid customer\_id should be provided.

**Test steps:**

1. Start Backend Server.
2. Send checkQueue get request with an invalid queue\_id.
3. Receive a 400 Bad request Failed response.
4. **Expected Result:** 400 Bad Request Failed response is displayed together with the error message: 'ID should be 10-digits'.

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 64
6 ETag: W/"40-11Tl6ZvHpWdnmc5N6+CEOjH9nyw"
7 Date: Fri, 27 Nov 2020 06:24:19 GMT
8 Connection: close
9
10 {
11   "error": "ID should be 10-digits",
12   "code": "INVALID_QUERY_STRING"
13 }
```

**Title:** Check Queue API – An unsuccessfully request does not allow customer to know the number of people in a queue when a non-existent customer\_id is provided.

**Description:** An unsuccessful request does not allow customer to know the number of people in a queue when a non-existent queue\_id is provided. Error message together with 404 Not Found should be displayed.

**Precondition:** A non-existent queue\_id, valid queue\_id, valid customer\_id.

**Test steps:**

1. Start Backend Server.
2. Send checkQueue get request with a non-existent queue\_id.
3. Receive a 404 Not Found response.
4. **Expected Result:** 404 not found is displayed together with the error message: "Queue Id: (queue\_id) Not Found".

**Example Evidence:**

```
1 HTTP/1.1 404 Not Found
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-KPAp4hZtnP9DABi1BOXX62EZ9TY"
7 Date: Fri, 27 Nov 2020 06:32:51 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12342 Not Found",
12   "code": "UNKNOWN_QUEUE"
13 }
```

## Customer: Arrival Rate

```
PASS tests/arrivalRate.test.js
  ✓ success (315 ms)
  ✓ error - queue id does not exists (312 ms)
  ✓ error - from not following format (6 ms)
  ✓ error - duration negative (7 ms)
  ✓ error - duration NaN (6 ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
```

**Title:** Arrival Rate API – A successful request to allow company to know the number of arrival within a certain period of time in buckets of 1 seconds.

**Description:** A successful request made to the server would display the count of customers arriving within a period when valid queue\_id, from and duration is entered.

**Precondition:** Valid queue\_id, duration and from should be entered

### Test steps:

1. Start Backend Server.
2. Send arrivalRate get request with a valid queue\_id, duration and from.
3. Receive a 200 OK response.
4. **Expected Result:** 200 ok success response is displayed count of people and timestamp.

**Title:** Arrival Rate API – An unsuccessful request to which does not allow the company to know the number of arrival within a certain period of time in buckets of 1 seconds due to unknown queue\_id.

**Description:** An unsuccessful request does not allow company to know the number of arrival in a given time when an unknown queue\_id is provided. Error message together with 404 Not Found should be displayed

**Precondition:** Unknown queue\_id, valid duration, valid from.

**Test steps:**

1. Start Backend Server.
2. Send arrivalRate get request with unknown queue\_id.
3. Receive a 404 Not Found response.
4. **Expected Result:** 404 not found is displayed together with the error message: "Queue Id: (queue\_id) Not Found".

**Example Evidence:**

```
1 HTTP/1.1 404 Not Found
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 65
6 ETag: W/"41-IsuL6NAq6g6YIVSMOJh+aemOFF8"
7 Date: Fri, 27 Nov 2020 07:18:07 GMT
8 Connection: close
9
10 {
11   "error": "Queue Id: QUEUE12344 Not Found",
12   "code": "UNKNOWN_QUEUE"
13 }
```

**Title:** Arrival Rate API – An unsuccessful request does not allow company to know the number of arrivals in a given time when an invalid queue\_id is provided.

**Description:** An unsuccessful request does not allow customer to know the number of people in a queue when a valid queue\_id is entered but an invalid customer\_id is provided. Error message together with 400 Bad Request should be displayed.

**Precondition:** Invalid queue\_id should be entered, valid duration, valid from.

**Test steps:**

1. Start Backend Server.
2. Send arrivalRate get request with an invalid queue\_id.
3. Receive a 400 Bad request response.
4. **Expected Result:** 400 Bad Request Failed response is displayed together with the error message: 'you have an invalid query string!'.

**Example Evidence:**

```
1 HTTP/1.1 400 Bad Request
2 X-Powered-By: Express
3 Access-Control-Allow-Origin: *
4 Content-Type: application/json; charset=utf-8
5 Content-Length: 75
6 ETag: W/"4b-g3Wtf570qQqip4uC3qbhJ42766E"
7 Date: Fri, 27 Nov 2020 07:24:34 GMT
8 Connection: close
9
10 {
11   "error": "You have an invalid query string!",
12   "code": "INVALID_QUERY_STRING"
13 }
```