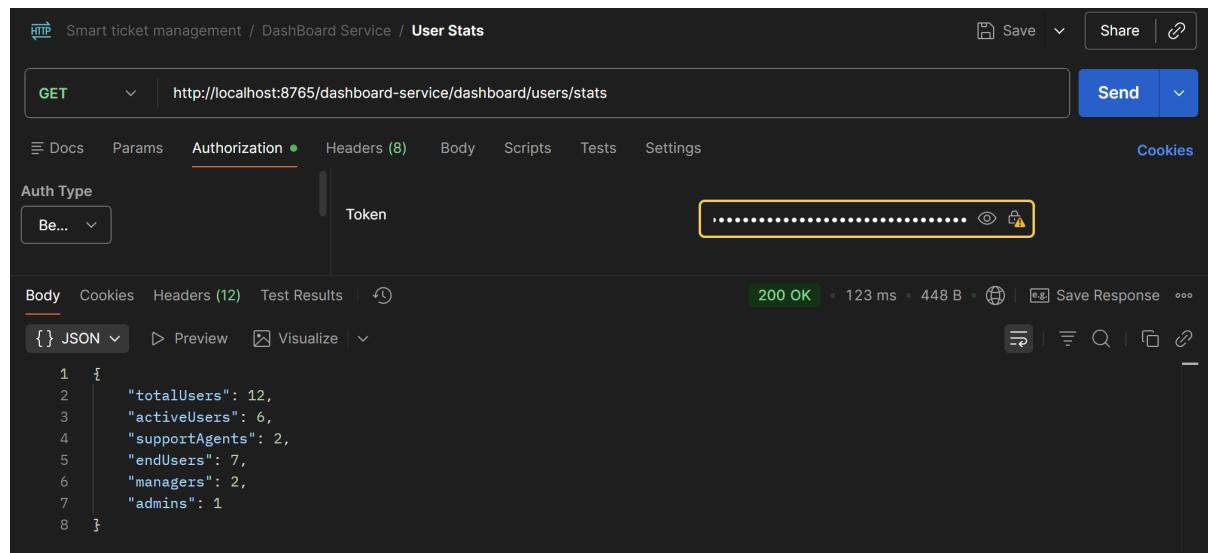


# API Tests

## DashBoard Service

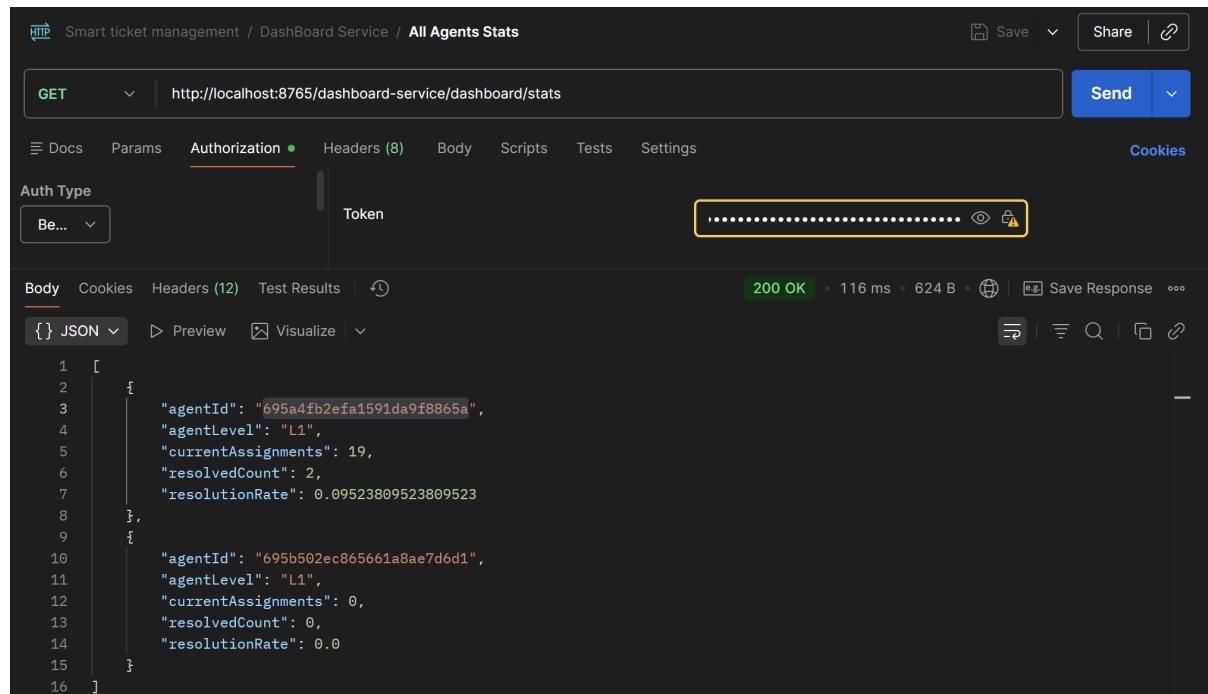
### 1. Total User Stats



Screenshot of the Postman interface showing a successful GET request to `http://localhost:8765/dashboard-service/dashboard/users/stats`. The response body is a JSON object containing user statistics:

```
1 {  
2   "totalUsers": 12,  
3   "activeUsers": 6,  
4   "supportAgents": 2,  
5   "endUsers": 7,  
6   "managers": 2,  
7   "admins": 1  
8 }
```

### 2. All Agents Stats



Screenshot of the Postman interface showing a successful GET request to `http://localhost:8765/dashboard-service/dashboard/stats`. The response body is a JSON array of agent statistics:

```
1 [  
2   {  
3     "agentId": "695a4fb2efab1591da9f8865a",  
4     "agentLevel": "L1",  
5     "currentAssignments": 19,  
6     "resolvedCount": 2,  
7     "resolutionRate": 0.09523809523809523  
8   },  
9   {  
10    "agentId": "695b502ec865661a8ae7d6d1",  
11    "agentLevel": "L1",  
12    "currentAssignments": 0,  
13    "resolvedCount": 0,  
14    "resolutionRate": 0.0  
15  }  
16 ]
```

### 3. Particular Agent Stats

The screenshot shows a dark-themed API testing interface. At the top, the URL is `http://localhost:8765/dashboard-service/dashboard/695a4fb2efa1591da9f8865a/stats`. The 'Authorization' tab is selected, showing 'Bearer Token' and a redacted token field. The 'Body' tab is selected, displaying a JSON response:

```
1 {  
2   "agentId": "695a4fb2efa1591da9f8865a",  
3   "agentLevel": "L1",  
4   "currentAssignments": 19,  
5   "resolvedCount": 2,  
6   "resolutionRate": 0.09523809523809523  
7 }
```

The status bar at the bottom indicates a **200 OK** response with **107 ms** latency and **496 B** size.

## 4. Ticket Status Summary Stats

The screenshot shows a dark-themed API testing interface. At the top, the URL is `http://localhost:8765/dashboard-service/dashboard/tickets/status-summary`. The 'Authorization' tab is selected, showing 'Bearer Token' and a redacted token field. The 'Body' tab is selected, displaying a JSON response:

```
1 [  
2   {  
3     "status": "OPEN",  
4     "count": 2  
5   },  
6   {  
7     "status": "ESCALATED",  
8     "count": 1  
9   }  
10 ]
```

The status bar at the bottom indicates a **200 OK** response with **445 ms** latency and **430 B** size.

## 5. Priority Status Summary

The screenshot shows a dark-themed API testing interface. At the top, the URL is `http://localhost:8765/dashboard-service/dashboard/tickets/status-priority-summary`. The 'Authorization' tab is selected, showing 'Bearer Token' and a redacted token field. The 'Body' tab is selected, displaying a JSON response:

```
1 [  
2   {  
3     "priority": "MEDIUM",  
4     "count": 1  
5   },  
6   {  
7     "priority": "HIGH",  
8     "count": 1  
9   },  
10  {  
11    "priority": "LOW",  
12    "count": 1  
13  }  
14 ]
```

The status bar at the bottom indicates a **200 OK** response with **141 ms** latency and **460 B** size.

## 6. Category Status Summary

The screenshot shows the Smart ticket management API dashboard with the URL `http://localhost:8765/dashboard-service/dashboard/tickets/category-summary`. The response is a 200 OK status with 91 ms latency and 473 B size. The JSON response body contains two categories:

```
1 [  
2 {  
3     "categoryId": "6959eef68c572994336fa567",  
4     "count": 1  
5 },  
6 {  
7     "categoryId": "6959eee08c572994336fa566",  
8     "count": 2  
9 }  
10 ]
```

## 7. Agent Summary Report

The screenshot shows the Smart ticket management API dashboard with the URL `http://localhost:8765/dashboard-service/dashboard/assignments/agent-summary`. The response is a 200 OK status with 464 ms latency and 664 B size. The JSON response body contains two agents:

```
1 [  
2 {  
3     "agentId": "695803d8a6738a0ce21b3baa",  
4     "assignedCount": 0,  
5     "resolvedCount": 0,  
6     "overdueCount": 0,  
7     "escalationLevel": 1,  
8     "averageResolutionTimeMinutes": 0.0  
9 },  
10 {  
11     "agentId": "695a4fb2efa1591da9f8865a",  
12     "assignedCount": 17,  
13     "resolvedCount": 0,  
14     "overdueCount": 0,  
15     "escalationLevel": 0,  
16     "averageResolutionTimeMinutes": 0.0  
17 }  
18 ]
```

## 8.Escalation Summary Report

The screenshot shows the Postman interface with the following details:

- URL:** `http://localhost:8765/dashboard-service/dashboard/assignments/escalation-summary`
- Method:** GET
- Headers:** Authorization (set to Token)
- Body:** JSON response (200 OK) showing a list of escalation levels and counts:

```
[{"level": 0, "count": 17}, {"level": 1, "count": 5}]
```

## 9 Ticket Status By User ID

The screenshot shows the Postman interface with the following details:

- URL:** `http://localhost:8765/dashboard-service/dashboard/tickets/user/69588735bedab262ae4d8d4f/stats`
- Method:** GET
- Headers:** Authorization (set to Bearer Token)
- Body:** JSON response (200 OK) showing ticket statistics for a user:

```
{"total": 3, "open": 2, "resolved": 0, "critical": 0}
```

## 10 Circuit Breaker

The screenshot shows the Postman interface with the following details:

- URL:** `http://localhost:8765/dashboard-service/actuator/circuitbreakers`
- Method:** GET
- Headers:** Authorization (set to Token)
- Body:** JSON response (200 OK) showing circuit breaker status:

```
{ "circuitBreakers": [ { "dashboardServiceCircuitBreaker": { "failureRate": "-1.0%", "slowCallRate": "-1.0%", "failureRateThreshold": "50.0%", "slowCallRateThreshold": "100.0%", "bufferedCalls": 2, "failedCalls": 0, "slowCalls": 0, "slowFailedCalls": 0, "notPermittedCalls": 0, "state": "CLOSED" } } ] }
```

## 11 Ticket Global Stats

The screenshot shows the Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:8765/dashboard-service/dashboard/tickets/global-stats
- Authorization:** Bearer Token (selected)
- Body:** JSON (selected) - displays the response body:

```
1 {  
2     "total": 3,  
3     "open": 2,  
4     "resolved": 0,  
5     "critical": 0  
6 }
```
- Response Status:** 200 OK
- Response Time:** 109 ms
- Response Size:** 406 B

## 12 Unauthorized Case

The screenshot shows the Postman interface with the following details:

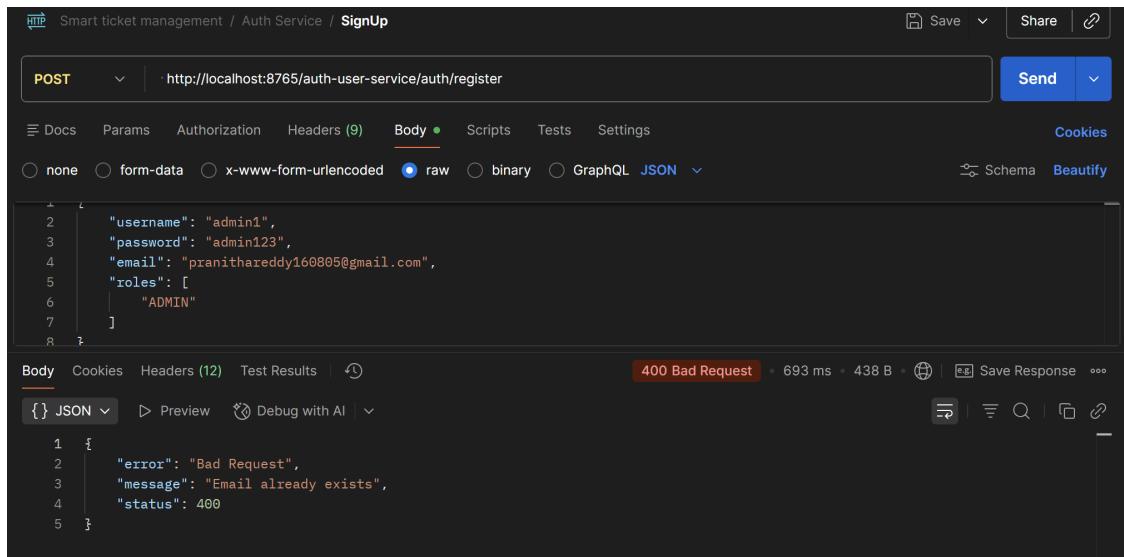
- Method:** GET
- URL:** http://localhost:8765/dashboard-service/dashboard/tickets/global-stats
- Authorization:** Bearer Token (selected)
- Body:** Raw (selected) - displays the response body:

```
1
```
- Response Status:** 401 Unauthorized
- Response Time:** 18 ms
- Response Size:** 407 B

A note in the response body says: "Pass the correct auth credentials".

# Auth-User Microservice

## Registering with already existing case



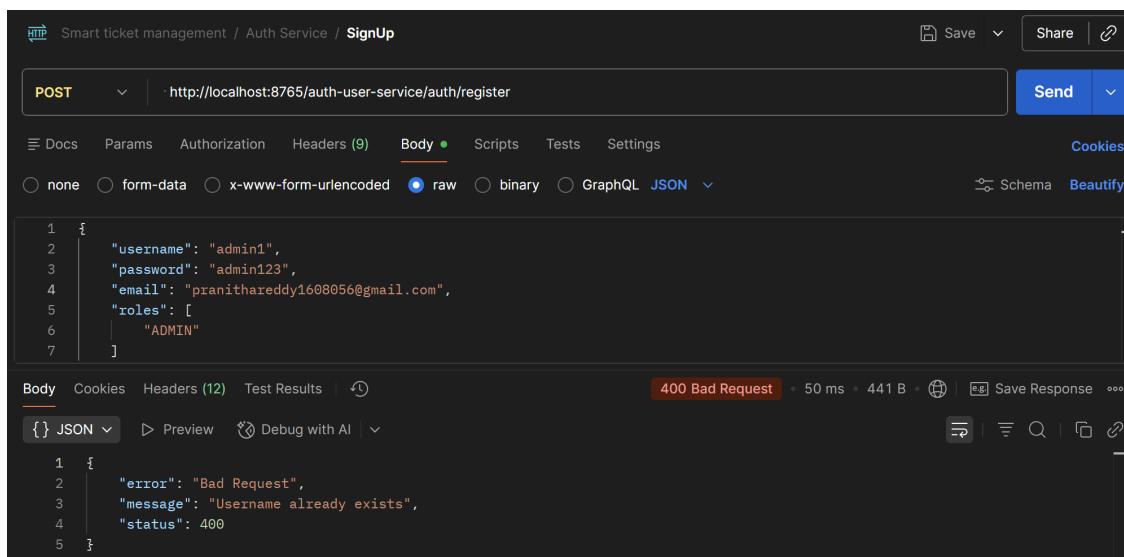
Smart ticket management / Auth Service / **SignUp**

POST <http://localhost:8765/auth-user-service/auth/register> Send

Body `{ "username": "admin1", "password": "admin123", "email": "pranithareddy160805@gmail.com", "roles": [ "ADMIN" ] }`

400 Bad Request 693 ms 438 B Save Response

`{ "error": "Bad Request", "message": "Email already exists", "status": 400 }`



Smart ticket management / Auth Service / **SignUp**

POST <http://localhost:8765/auth-user-service/auth/register> Send

Body `{ "username": "admin1", "password": "admin123", "email": "pranithareddy160805@gmail.com", "roles": [ "ADMIN" ] }`

400 Bad Request 50 ms 441 B Save Response

`{ "error": "Bad Request", "message": "Username already exists", "status": 400 }`

## SignUp successful

The screenshot shows a POST request to `http://localhost:8765/auth-user-service/auth/register`. The request body is raw JSON:

```
1 {
2     "username": "admin12",
3     "password": "admin123",
4     "email": "pranithareddy@ticket.com",
5     "roles": [
6         "ADMIN"
7     ]
}
```

The response status is 201 Created, with a response time of 728 ms and a response size of 419 B. The response body is:

```
1 {
2     "success": true,
3     "message": "User created successfully"
4 }
```

## SignIn successful

The screenshot shows a POST request to `http://localhost:8765/auth-user-service/auth/login`. The request body is raw JSON:

```
1 {
2     "email": "pranithareddy@ticket.com",
3     "password": "admin123"
4 }
```

The response status is 200 OK, with a response time of 170 ms and a response size of 645 B. The response body is:

```
1 {
2     "success": true,
3     "message": "eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiI2OTVkNzMzYwMzM4OTMyY2UiLCJlbWFpbCI6InByYW5pdGhcmVkJH1AdGlja2V0LmNvbSIsInVzZXJuYWlIjoiYWRTaW4xMiIsInJvbGVzIjpBIkFETU10I10sIm1hdCI6MTc2NczMTU4OCwiZXhwIjoxNzY3NzM1MTg4fQ.KORWNxj7T0gPUsSmg1EFLNIP7c-9wMMwXb6JUU4AE-c"
4 }
```

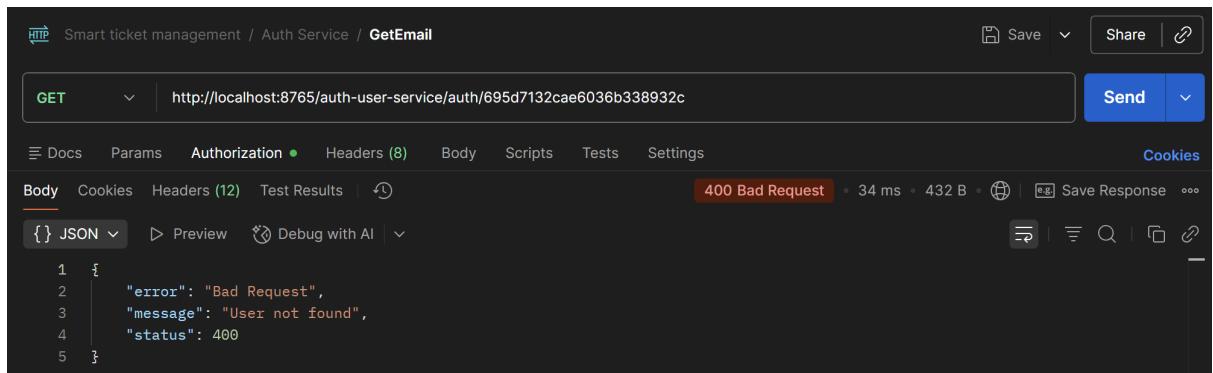
## Get user Email by ID

The screenshot shows a GET request to `http://localhost:8765/auth-user-service/auth/695d7132cae6036b338932ce`. The request includes an Authorization header set to Bearer Token with a token value.

The response status is 200 OK, with a response time of 209 ms and a response size of 392 B. The response body is:

```
1 pranithareddy@ticket.com
```

## Get user Email by ID failure if ID does not exists



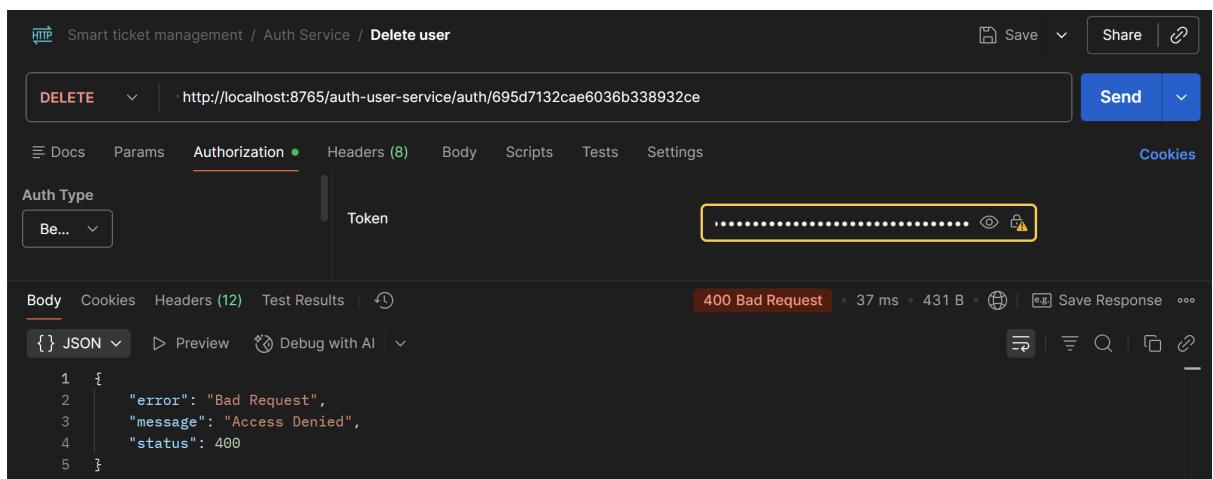
Smart ticket management / Auth Service / GetEmail

GET http://localhost:8765/auth-user-service/auth/695d7132cae6036b338932c

400 Bad Request 34 ms 432 B

```
{ "error": "Bad Request", "message": "User not found", "status": 400 }
```

## When User, Agent, Manager tries to delete



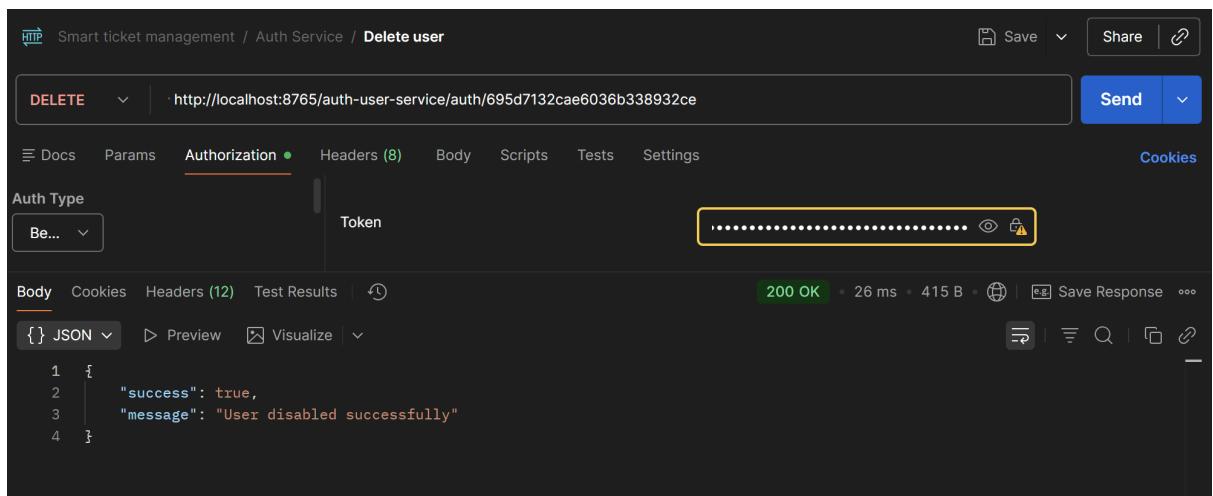
Smart ticket management / Auth Service / Delete user

DELETE http://localhost:8765/auth-user-service/auth/695d7132cae6036b338932ce

400 Bad Request 37 ms 431 B

```
{ "error": "Bad Request", "message": "Access Denied", "status": 400 }
```

## When Admin deletes



Smart ticket management / Auth Service / Delete user

DELETE http://localhost:8765/auth-user-service/auth/695d7132cae6036b338932ce

200 OK 26 ms 415 B

```
{ "success": true, "message": "User disabled successfully" }
```

## With Wrong Token

The screenshot shows a Postman request for 'Get all Users' (GET http://localhost:8765/auth-user-service/users). The 'Authorization' tab is selected, showing a 'Token' field with a placeholder 'Be...'. The response status is '401 Unauthorized' with a message 'Pass the correct auth credentials'. The response body is empty.

## Get all users

The screenshot shows a Postman request for 'Get all Users' (GET http://localhost:8765/auth-user-service/users). The 'Authorization' tab is selected, showing a 'Token' field with a placeholder 'Be...'. The response status is '200 OK' with a message 'Visualize'. The response body is a JSON array containing one user object:

```
1 [  
2 {  
3   "id": "695d1f1093b428245d583ebb",  
4   "displayName": "USR-695D1F",  
5   "email": "pranithareddy160805@gmail.com",  
6   "username": "admin1",  
7   "enabled": true,  
8   "roles": [  
9     "ADMIN"  
10   ],  
11   "agentProfile": null  
]
```

## User not found case

The screenshot shows a Postman request for 'Get User By ID' (GET http://localhost:8765/auth-user-service/users/6950dc147dae187e1ce755df). The 'Authorization' tab is selected, showing a 'Bearer Token' field with a placeholder 'Be...'. The response status is '400 Bad Request' with a message 'Debug with AI'. The response body is a JSON object:

```
1 {  
2   "error": "Bad Request",  
3   "message": "User not found",  
4   "status": 400  
5 }
```

## Get user by Email

The screenshot shows the API interface for the 'Auth Service'. A GET request is made to `http://localhost:8765/auth-user-service/users/email/varakanthampranitha@gmail.com`. The 'Authorization' tab is selected, showing a 'Bearer Token' input field containing a long token. The response is a 200 OK status with a response time of 44 ms, a body size of 529 B, and a save response button.

```
1 {  
2   "id": "695d30431414c64f505922d7",  
3   "displayId": "USR-695D30",  
4   "email": "varakanthampranitha@gmail.com",  
5   "username": "user",  
6   "enabled": true,  
7   "roles": [  
8     "USER"  
9   ],  
10  "agentProfile": null  
11 }
```

## Update user by ID

The screenshot shows the API interface for the 'Auth Service'. A PUT request is made to `http://localhost:8765/auth-user-service/users/695d1f1093b428245d583ebb`. The 'Body' tab is selected, showing raw JSON input with fields: 'username': 'pranitha12', 'email': 'pranithareddy160805@gmail.com', and 'roles': ['ADMIN']. The response is a 200 OK status with a response time of 34 ms, a body size of 414 B, and a save response button.

```
1 {  
2   "username": "pranitha12",  
3   "email": "pranithareddy160805@gmail.com",  
4   "roles": [ "ADMIN" ]  
5 }
```

```
1 {  
2   "success": true,  
3   "message": "User updated successfully"  
4 }
```

## Creating (Agent,User,Manager) by ADMIN

The screenshot shows the API interface for the 'Auth Service'. A POST request is made to `http://localhost:8765/auth-user-service/create`. The 'Body' tab is selected, showing raw JSON input with fields: 'username': 'user1', 'password': 'user', 'email': 'pranithareddy160805@ticket.com', and 'roles': ['USER']. The response is a 200 OK status with a response time of 132 ms, a body size of 414 B, and a save response button.

```
1 {  
2   "username": "user1",  
3   "password": "user",  
4   "email": "pranithareddy160805@ticket.com",  
5   "roles": [  
6     "USER"  
7   ]  
8 }
```

```
1 {  
2   "success": true,  
3   "message": "User created successfully"  
4 }
```

## Increment Agent Assignment count

The screenshot shows a Postman interface for a PUT request to `http://localhost:8765/auth-user-service/internal/agents/695d24051414c64f505922d6/increment-assignments`. The Authorization tab is selected, showing a Bearer Token. The response status is 200 OK with a message: "Agent assignment count incremented".

```
1 {  
2   "success": true,  
3   "message": "Agent assignment count incremented"  
4 }
```

## Increment Resolved Count

The screenshot shows a Postman interface for a PUT request to `http://localhost:8765/auth-user-service/agents/695d24051414c64f505922d6/resolved`. The Authorization tab is selected, showing a Bearer Token. The response status is 200 OK with a message: "Agent resolved count incremented".

```
1 {  
2   "success": true,  
3   "message": "Agent resolved count incremented"  
4 }
```

## Increment Escalated Count

The screenshot shows a Postman interface for a PUT request to `http://localhost:8765/auth-user-service/agents/695d24051414c64f505922d6/unresolved`. The Authorization tab is selected, showing a Bearer Token. The response status is 200 OK with a message: "Agent escalated count increment".

```
1 {  
2   "success": true,  
3   "message": "Agent escalated count increment"  
4 }
```

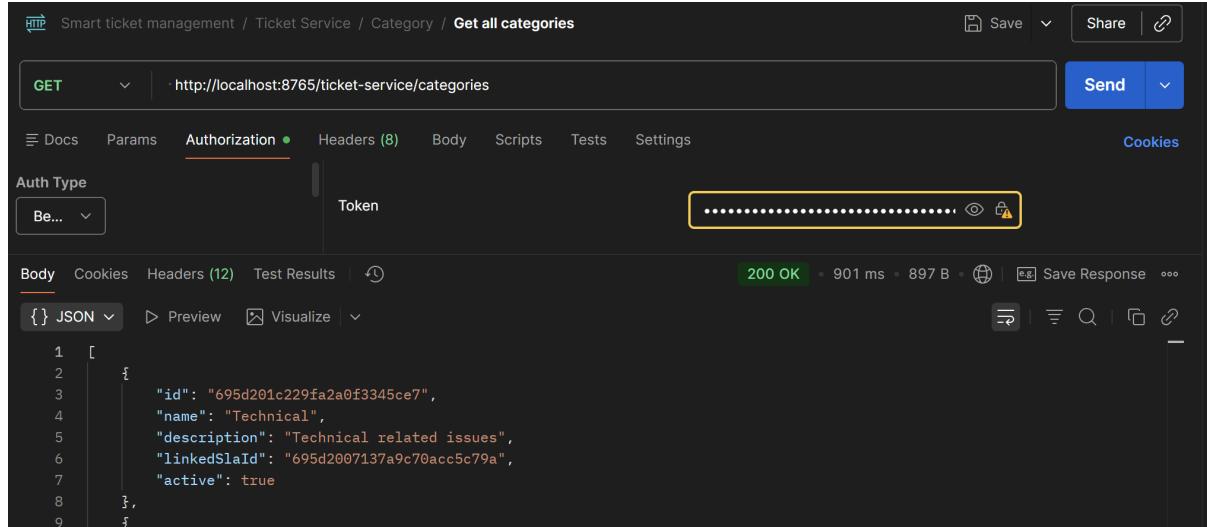
## Agents By Category

The screenshot shows a Postman interface for a GET request to `http://localhost:8765/auth-user-service/internal/agents?category=695d33fb7a48b3d67c0ba8a8`. The Authorization tab is selected, showing a Bearer Token. The response status is 200 OK with an empty JSON array.

```
1 []
```

# Ticket MicroService

## Get all Categories



Smart ticket management / Ticket Service / Category / Get all categories

GET http://localhost:8765/ticket-service/categories

Auth Type: Be...

Token: [REDACTED]

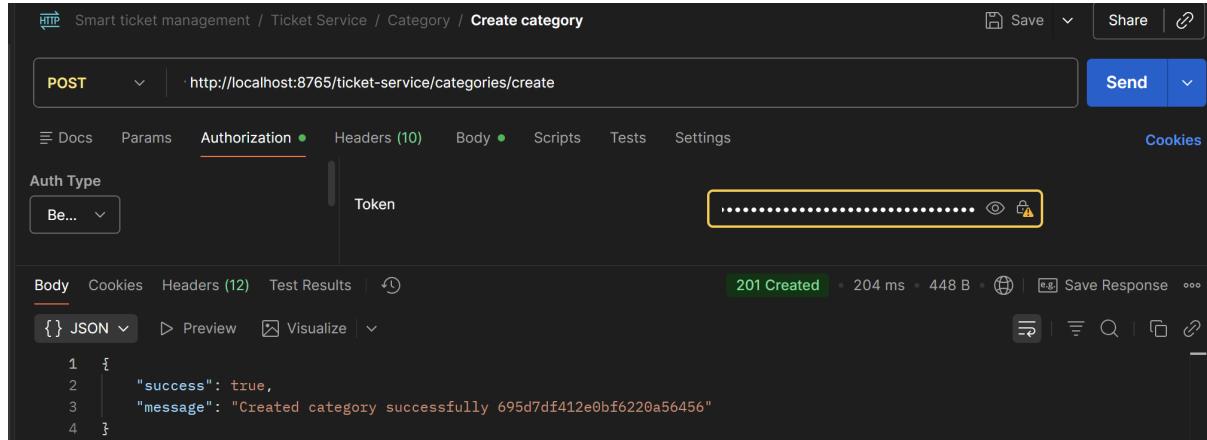
Body: { } JSON

Headers (12)

200 OK 901 ms 897 B

```
[{"id": "695d201c229fa2a0f3345ce7", "name": "Technical", "description": "Technical related issues", "linkedSlaId": "695d2007137a9c70acc5c79a", "active": true}]
```

## Create Category Only Admin can Add



Smart ticket management / Ticket Service / Category / Create category

POST http://localhost:8765/ticket-service/categories/create

Auth Type: Be...

Token: [REDACTED]

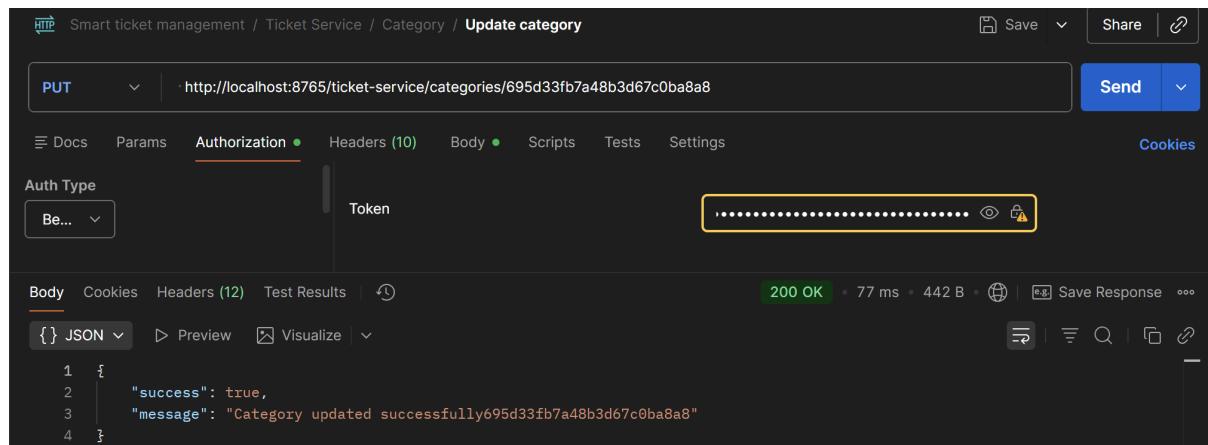
Body: { } JSON

Headers (12)

201 Created 204 ms 448 B

```
{"success": true, "message": "Created category successfully 695d7df412e0bf6220a56456"}
```

## Update Category Only By Admin



Smart ticket management / Ticket Service / Category / Update category

PUT http://localhost:8765/ticket-service/categories/695d33fb7a48b3d67c0ba8a8

Auth Type: Be...

Token: [REDACTED]

Body: { } JSON

Headers (12)

200 OK 77 ms 442 B

```
{"success": true, "message": "Category updated successfully 695d33fb7a48b3d67c0ba8a8"}
```

## Get Category By ID

The screenshot shows a Postman interface with the following details:

- Method:** GET
- URL:** <http://localhost:8765/ticket-service/categories/internal/695d33fb7a48b3d67c0ba8a8>
- Authorization:** Token (represented by a redacted token)
- Body:** JSON response (200 OK, 21 ms, 553 B)
  - id: "695d33fb7a48b3d67c0ba8a8",  
name: "Hardware and Kernel Issues",  
description: "Tickets related to hardware failures like laptops, desktops, printers",  
linkedSlaId: "sla101",  
active: true

## Delete category

Cannot delete category until the ticket is reassigned to another category

The screenshot shows a Postman interface with the following details:

- Method:** DELETE
- URL:** <http://localhost:8765/ticket-service/categories/695d33fb7a48b3d67c0ba8a8>
- Authorization:** Bearer Token (represented by a redacted token)
- Body:** JSON response (400 Bad Request, 66 ms, 450 B)
  - error: "Bad Request",  
message: "Tickets exist, reassign required",  
status: 400

## Ticket Activity Based on ID

The screenshot shows a Postman interface with the following details:

- Method:** GET
- URL:** <http://localhost:8765/ticket-service/tickets/695d3b217a48b3d67c0ba8af/activity>
- Authorization:** Bearer Token (represented by a redacted token)
- Body:** JSON response (200 OK, 53 ms, 567 B)
  - [
    - {  
id: "695d3b217a48b3d67c0ba8b0",  
ticketId: "695d3b217a48b3d67c0ba8af",  
actorId: "695d30431414c64f505922d7",  
actionType: "CREATED",  
details: "Ticket CREATED",  
timestamp: "2026-01-06T16:41:05.519Z"

## Internal Log Activity

The screenshot shows a Postman request for "Log activity". The URL is `http://localhost:8765/ticket-service/tickets/695d3b217a48b3d67c0ba8af/activity/internal/comment?actorId=695d24051414c64f...`. The response status is 200 OK with a timestamp of 2026-01-06T21:35:38.055415824Z. The request body is a JSON object:

```
1 {  
2   "id": "695d802a12e0bf6220a56457",  
3   "ticketId": "695d3b217a48b3d67c0ba8af",  
4   "actorId": "695d24051414c64f505922d6",  
5   "actionType": "COMMENT",  
6   "details": "Investigating",  
7   "timestamp": "2026-01-06T21:35:38.055415824Z"  
8 }
```

## Only User can add Ticket (Case when admin tried to add ticket)

The screenshot shows a Postman request for "Create Ticket". The URL is `http://localhost:8765/ticket-service/tickets/create`. The response status is 400 Bad Request. The error message in the response body is:

```
1 {  
2   "error": "Bad Request",  
3   "message": "Access Denied",  
4   "status": 400  
5 }
```

## Close Ticket when ticket not found

The screenshot shows a Postman request for "Close Ticket". The URL is `http://localhost:8765/ticket-service/tickets/69510383bbb3e37045da04ad/close`. The response status is 400 Bad Request. The error message in the response body is:

```
1 {  
2   "error": "Bad Request",  
3   "message": "Ticket not found",  
4   "status": 400  
5 }
```

## Closing condition violation

The screenshot shows a REST API testing interface. The URL is `http://localhost:8765/ticket-service/tickets/695d32d27a48b3d67c0ba8a5/close`. The response status is **400 Bad Request** with a message: "Ticket must be assigned before closing".

```
1 {
2     "error": "Bad Request",
3     "message": "Ticket must be assigned before closing",
4     "status": 400
5 }
```

## Reopen Ticket Violation

The screenshot shows a REST API testing interface. The URL is `http://localhost:8765/ticket-service/tickets/695d32d27a48b3d67c0ba8a5/reopen`. The response status is **400 Bad Request** with a message: "Only RESOLVED or CLOSED tickets can be reopened".

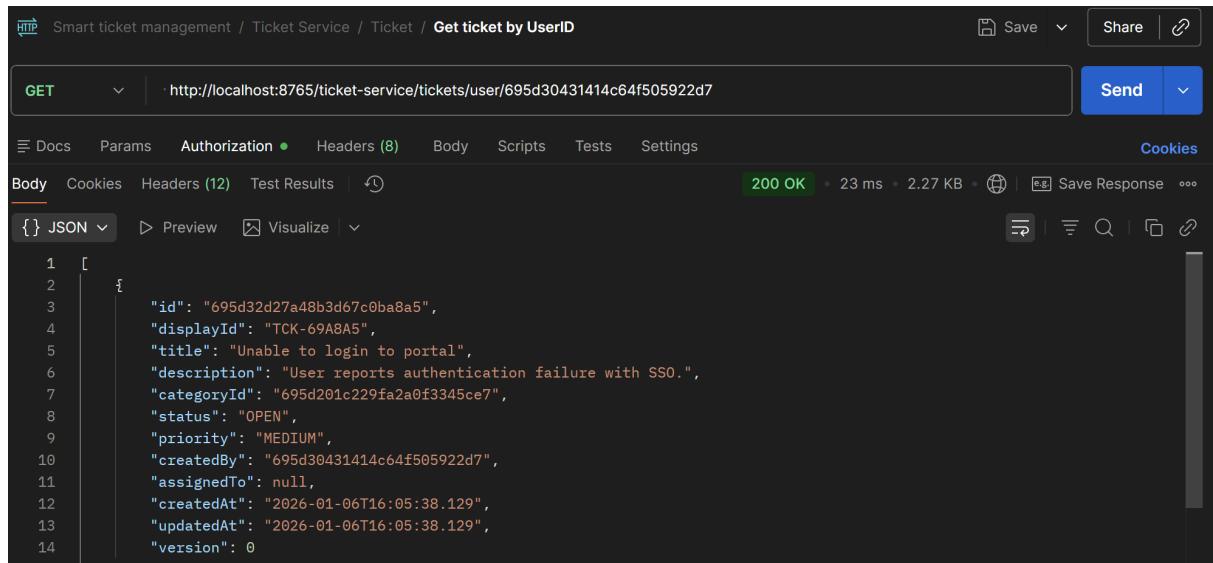
```
1 {
2     "error": "Bad Request",
3     "message": "Only RESOLVED or CLOSED tickets can be reopened",
4     "status": 400
5 }
```

## Recent Tickets

The screenshot shows a REST API testing interface. The URL is `http://localhost:8765/ticket-service/tickets/recent`. The response status is **200 OK** with a message: "Success".

```
1 [
2     {
3         "id": "695d3b217a48b3d67c0ba8af",
4         "displayId": "TCK-69A8AF",
5         "title": "Software",
6         "description": "Application bugs, crashes, and installation errors.",
7         "categoryId": "695d33d17a48b3d67c0ba8a7",
8         "status": "OPEN",
9         "priority": "MEDIUM",
10        "createdBy": "695d30431414c64f505922d7",
11        "assignedTo": null,
12        "createdAt": "2026-01-06T16:41:05.512",
13        "updatedAt": "2026-01-06T16:41:05.512",
14        "version": 0
15    }
]
```

## Tickets Based on UserId



HTTP Smart ticket management / Ticket Service / Ticket / Get ticket by UserID

GET http://localhost:8765/ticket-service/tickets/user/695d30431414c64f505922d7

Send

Docs Params Authorization Headers (8) Body Scripts Tests Settings Cookies

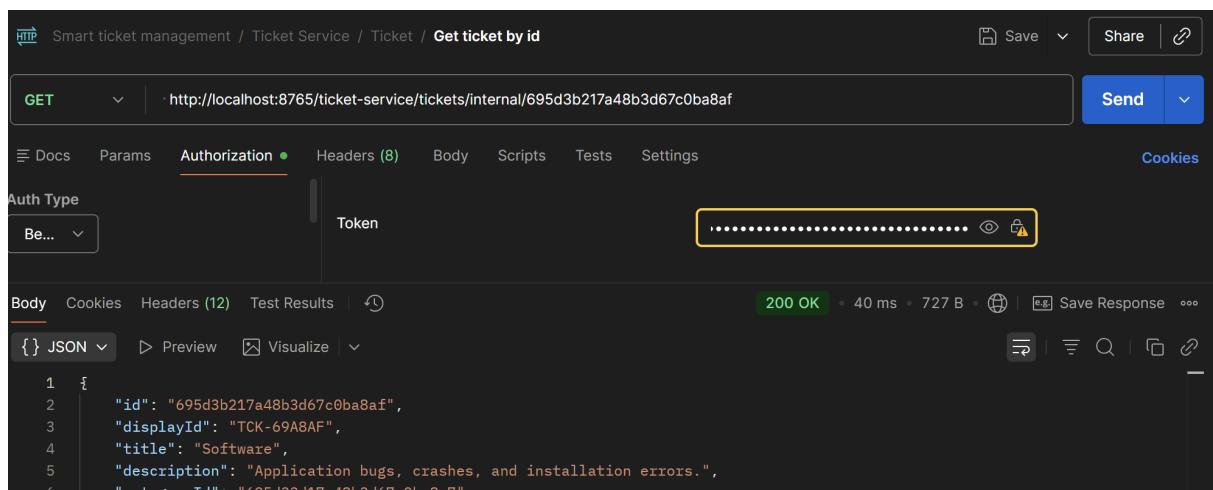
Body Cookies Headers (12) Test Results

200 OK • 23 ms • 2.27 KB • Save Response

{ } JSON Preview Visualize

```
1 [  
2 {  
3   "id": "695d32d27a48b3d67c0ba8a5",  
4   "displayId": "TCK-69A8A5",  
5   "title": "Unable to login to portal",  
6   "description": "User reports authentication failure with SSO.",  
7   "categoryId": "695d201c229fa2a0f3345ce7",  
8   "status": "OPEN",  
9   "priority": "MEDIUM",  
10  "createdBy": "695d30431414c64f505922d7",  
11  "assignedTo": null,  
12  "createdAt": "2026-01-06T16:05:38.129",  
13  "updatedAt": "2026-01-06T16:05:38.129",  
14  "version": 0  
15 ]
```

## Get Ticket By Id



HTTP Smart ticket management / Ticket Service / Ticket / Get ticket by id

GET http://localhost:8765/ticket-service/tickets/internal/695d3b217a48b3d67c0ba8af

Send

Docs Params Authorization Headers (8) Body Scripts Tests Settings Cookies

Auth Type Be... Token

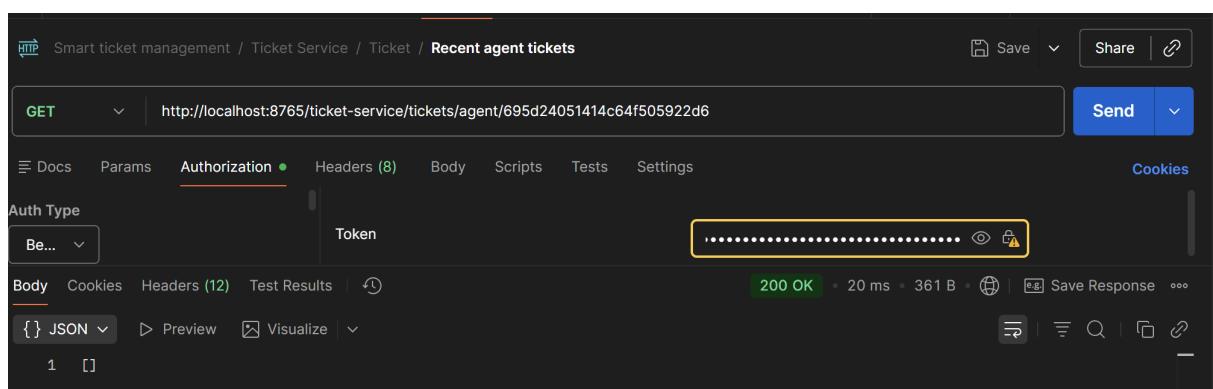
Body Cookies Headers (12) Test Results

200 OK • 40 ms • 727 B • Save Response

{ } JSON Preview Visualize

```
1 {  
2   "id": "695d3b217a48b3d67c0ba8af",  
3   "displayId": "TCK-69A8AF",  
4   "title": "Software",  
5   "description": "Application bugs, crashes, and installation errors.",  
6   "categoryId": "695d33d17a48b3d67c0ba8a7"  
7 }
```

## Recent Agent Tickets



HTTP Smart ticket management / Ticket Service / Ticket / Recent agent tickets

GET http://localhost:8765/ticket-service/tickets/agent/695d24051414c64f505922d6

Send

Docs Params Authorization Headers (8) Body Scripts Tests Settings Cookies

Auth Type Be... Token

Body Cookies Headers (12) Test Results

200 OK • 20 ms • 361 B • Save Response

{ } JSON Preview Visualize

```
1 [ ]
```

## Manual Assignment

HTTP Smart ticket management / Assignment-Escalation Service / Assignment / **Manual Assignment**

POST http://localhost:8765/assignment-escalation-service/assignments/manual?ticketId=695d32d27a48b3d67c0ba8a5&agentId=695... Save Share ⌂

Docs Params Authorization Headers (9) Body Scripts Tests Settings Cookies

Auth Type Be... Token

Body Cookies Headers (12) Test Results ⌂

400 Bad Request 55 ms 456 B ⌂ Save Response ⌂

{ } JSON Preview Debug with AI ⌂

```
1 {
2   "error": "Bad Request",
3   "message": "Ticket is already assigned to an agent",
4   "status": 400
5 }
```

## Making assignment Complete

HTTP Smart ticket management / Assignment-Escalation Service / Assignment / **Make assignment complete**

PUT http://localhost:8765/assignment-escalation-service/assignments/695d39e79e59db0397a258d4/complete Save Share ⌂

Docs Params Authorization Headers (9) Body Scripts Tests Settings Cookies

Auth Type Be... Token

Body Cookies Headers (12) Test Results ⌂

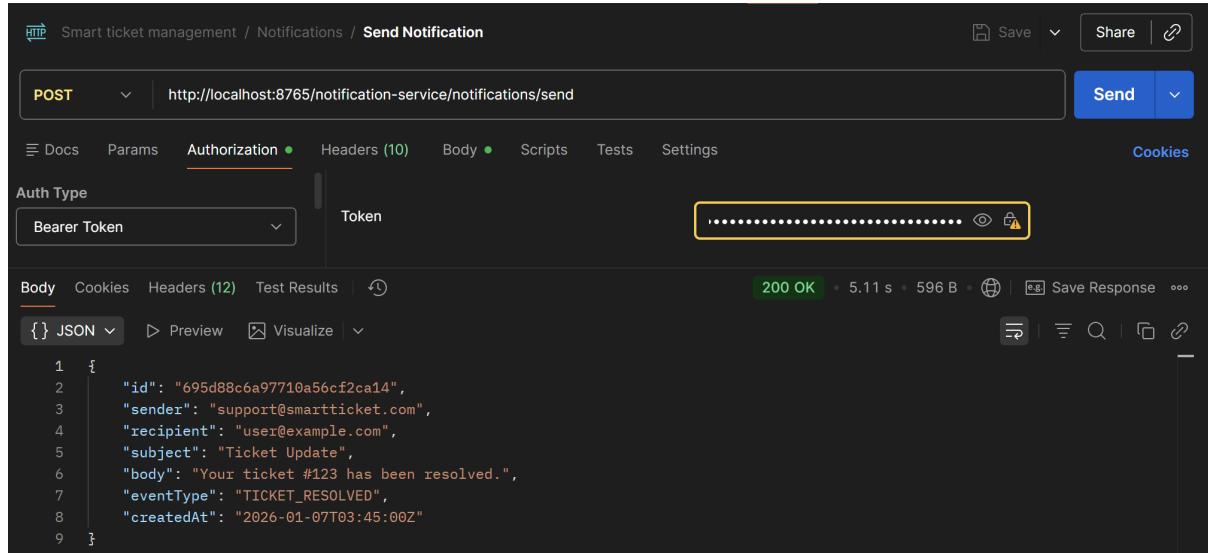
400 Bad Request 21 ms 438 B ⌂ Save Response ⌂

{ } JSON Preview Debug with AI ⌂

```
1 {
2   "error": "Bad Request",
3   "message": "Assignment not found",
4   "status": 400
5 }
```

# Notification Microservice

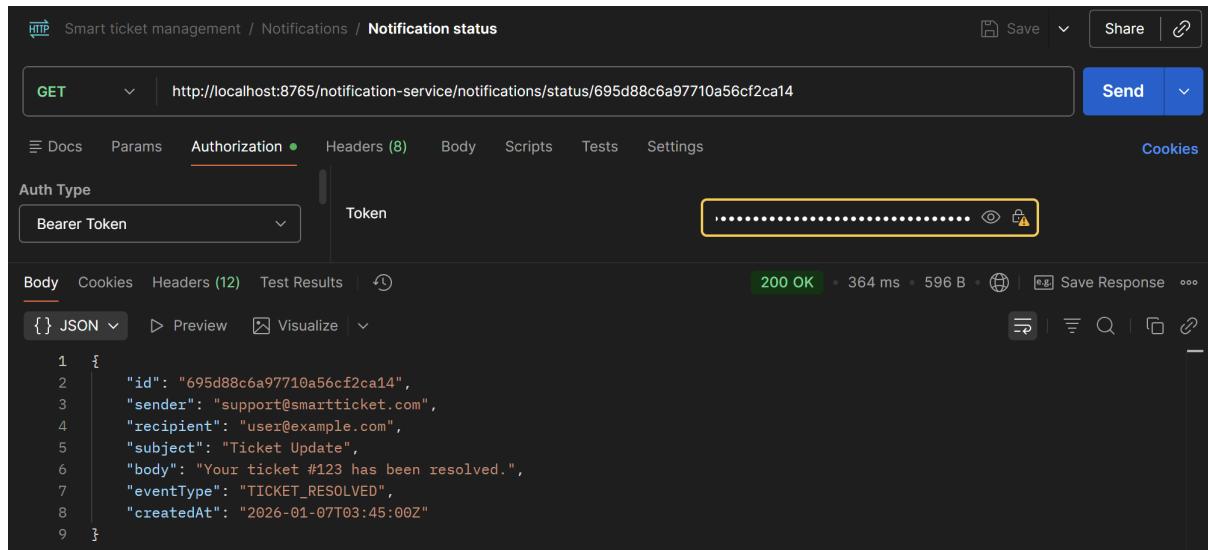
## Send Notification



A screenshot of the Postman application interface. The URL is `http://localhost:8765/notification-service/notifications/send`. The response status is `200 OK` with a response time of 5.11 s and a body size of 596 B. The response body is a JSON object:

```
1 {  
2   "id": "695d88c6a97710a56cf2ca14",  
3   "sender": "support@smartticket.com",  
4   "recipient": "user@example.com",  
5   "subject": "Ticket Update",  
6   "body": "Your ticket #123 has been resolved.",  
7   "eventType": "TICKET_RESOLVED",  
8   "createdAt": "2026-01-07T03:45:00Z"  
9 }
```

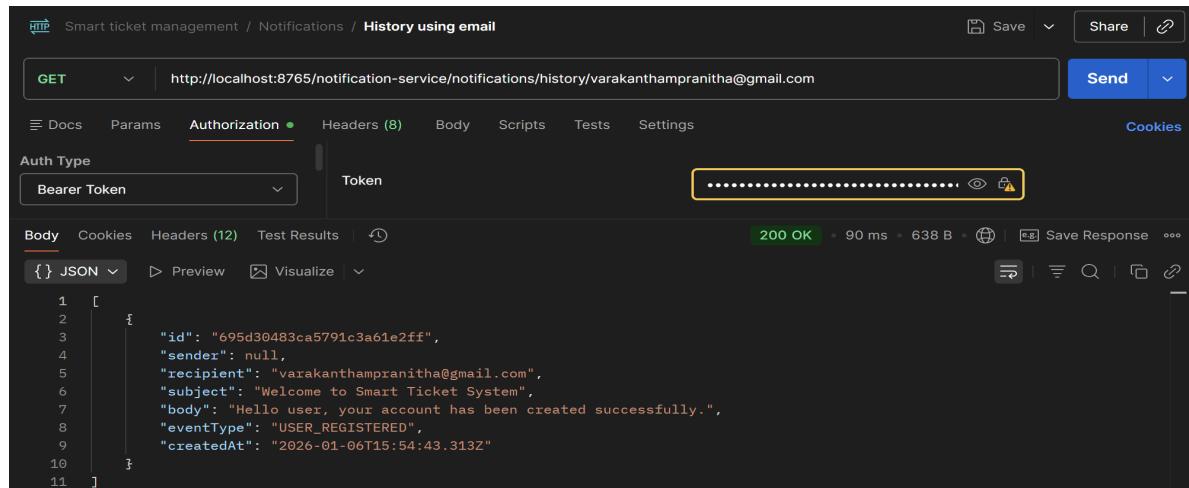
## Notification Status



A screenshot of the Postman application interface. The URL is `http://localhost:8765/notification-service/status/695d88c6a97710a56cf2ca14`. The response status is `200 OK` with a response time of 364 ms and a body size of 596 B. The response body is a JSON object:

```
1 {  
2   "id": "695d88c6a97710a56cf2ca14",  
3   "sender": "support@smartticket.com",  
4   "recipient": "user@example.com",  
5   "subject": "Ticket Update",  
6   "body": "Your ticket #123 has been resolved.",  
7   "eventType": "TICKET_RESOLVED",  
8   "createdAt": "2026-01-07T03:45:00Z"  
9 }
```

## Notifications by email



A screenshot of the Postman application interface. The URL is `http://localhost:8765/notification-service/notifications/history/varakanthampranitha@gmail.com`. The response status is `200 OK` with a response time of 90 ms and a body size of 638 B. The response body is a JSON array:

```
1 [  
2   {  
3     "id": "695d30483ca5791c3a61e2ff",  
4     "sender": null,  
5     "recipient": "varakanthampranitha@gmail.com",  
6     "subject": "Welcome to Smart Ticket System",  
7     "body": "Hello user, your account has been created successfully.",  
8     "eventType": "USER_REGISTERED",  
9     "createdAt": "2026-01-06T15:54:43.313Z"  
10    }  
11 ]
```