Serverless Chat Application Using WebSockets and API Gateway

Project Overview

The **Serverless Chat Application** is a real-time communication system built using AWS's serverless architecture. It allows users to connect, send, and receive messages over WebSocket connections. The system is scalable, fault-tolerant, and cost-efficient, leveraging various AWS services.

Architecture

Services Used:

1. **Amazon API Gateway**: Handles WebSocket communication.
2. **AWS Lambda**: Processes connection events and message transmissions.
3. **Amazon DynamoDB**: Stores user connection details and chat messages.
4. **Amazon CloudWatch**: Monitors connection metrics and logs.

Features

- Real-time messaging using WebSocket protocol.
- Persistent storage of messages and connection states.
- Serverless design ensures scalability and reduced management overhead.

System Design

API Gateway Routes:

1. **$connect**: Triggered when a client connects.
    - Stores connection ID in DynamoDB.
2. **$disconnect**: Triggered when a client disconnects.
    - Removes the connection ID from DynamoDB.
3. **sendmessage**: Handles user messages.

- Stores the message in DynamoDB.
- Routes messages to all connected clients.

Implementation Steps

1. Create the WebSocket API

- Open the **API Gateway Console** and select **WebSocket API**.
- Define routes for `$connect`, `$disconnect`, and `sendmessage`.
- Attach **Lambda functions** to handle these routes.

2. Deploy Lambda Functions

- Write Python scripts for the routes using the AWS SDK

3. Configure DynamoDB Tables

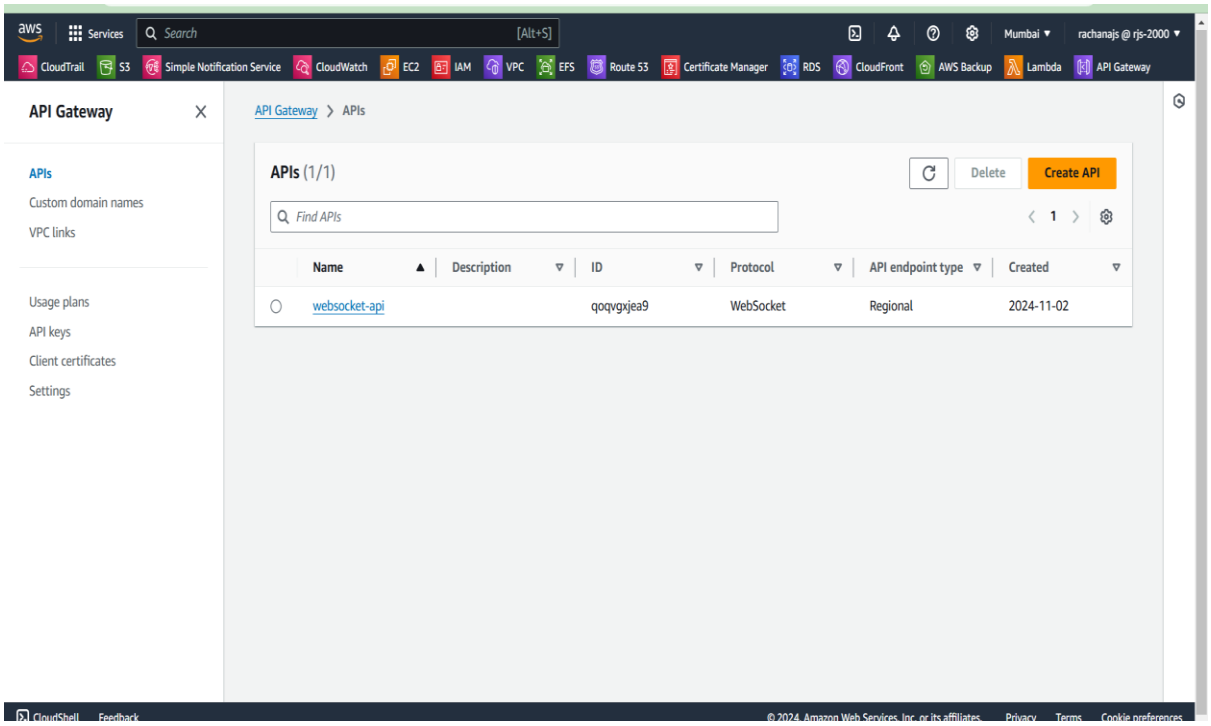
4.Set Up CloudWatch Monitoring

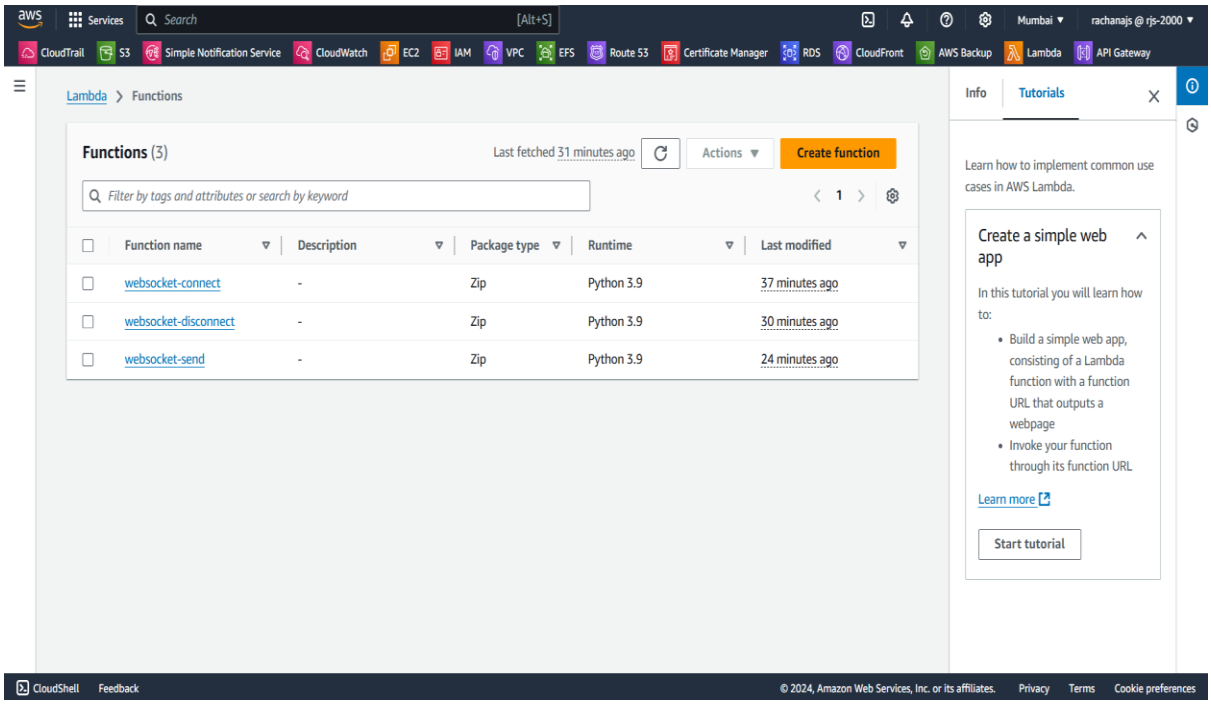
Testing and Deployment

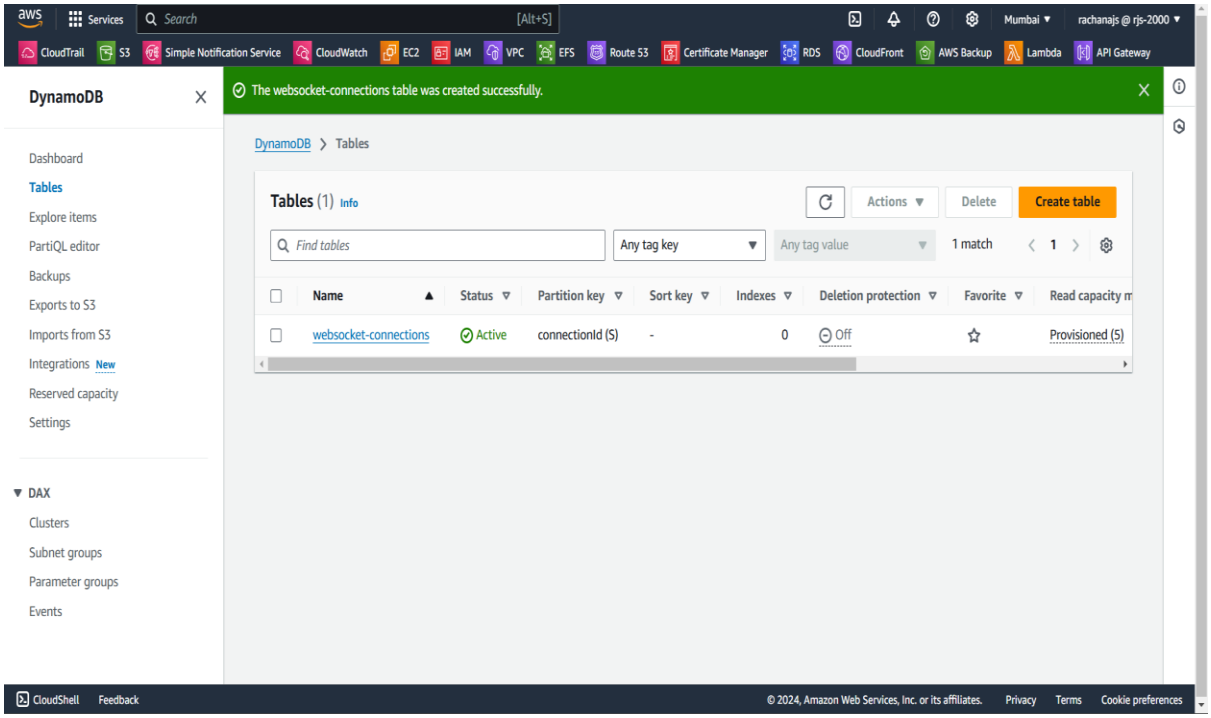
Using Postman for WebSocket Testing

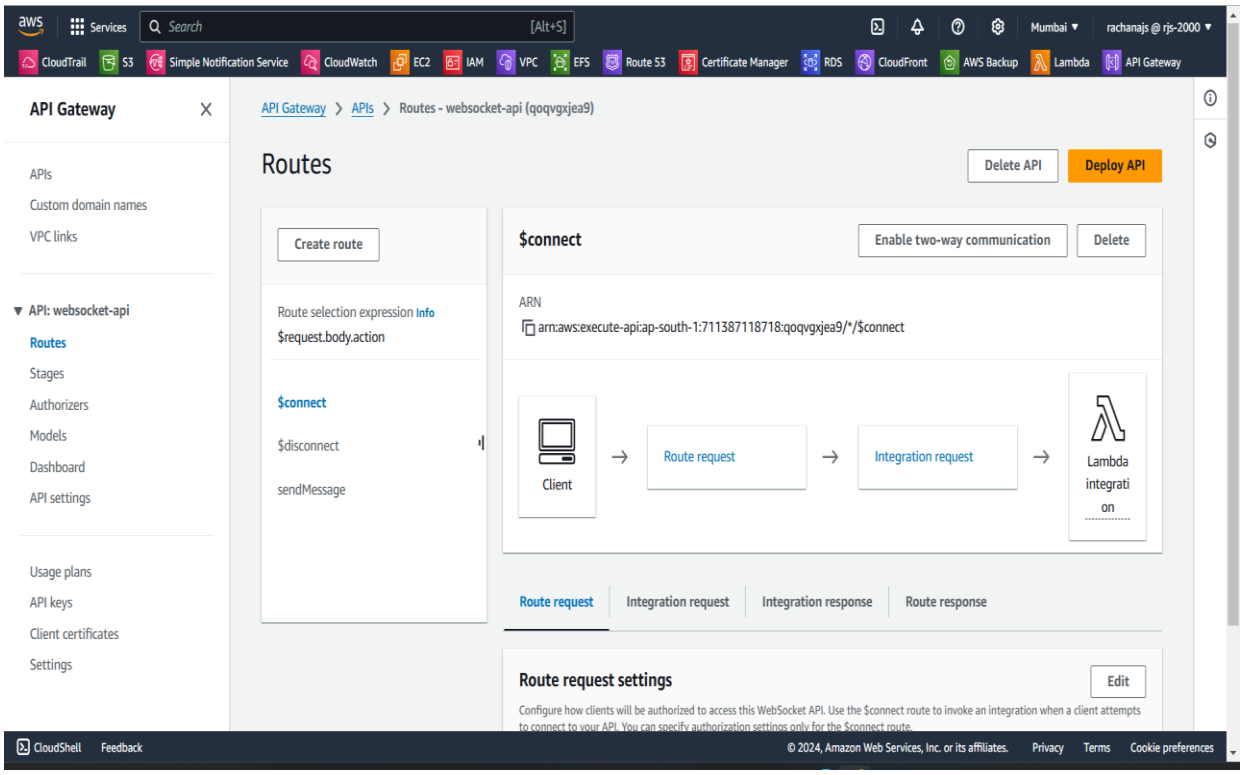
1. Open **Postman** and choose **New WebSocket Request**.
2. Use the API Gateway WebSocket URL (e.g., `wss://<api-id>.execute-api.<region>.amazonaws.com/<stage>`).
3. Test events like `$connect` and `sendmessage` by sending JSON payloads.

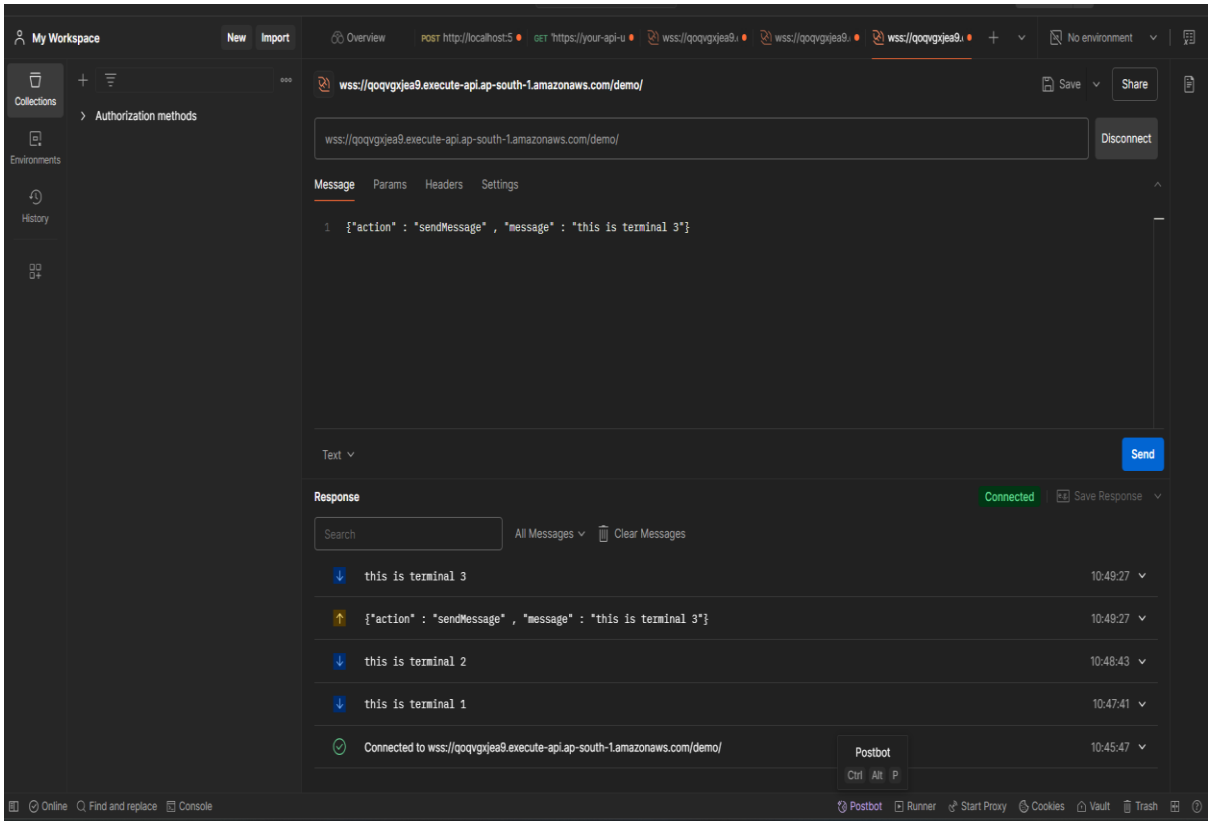
SCREENSHOTS:

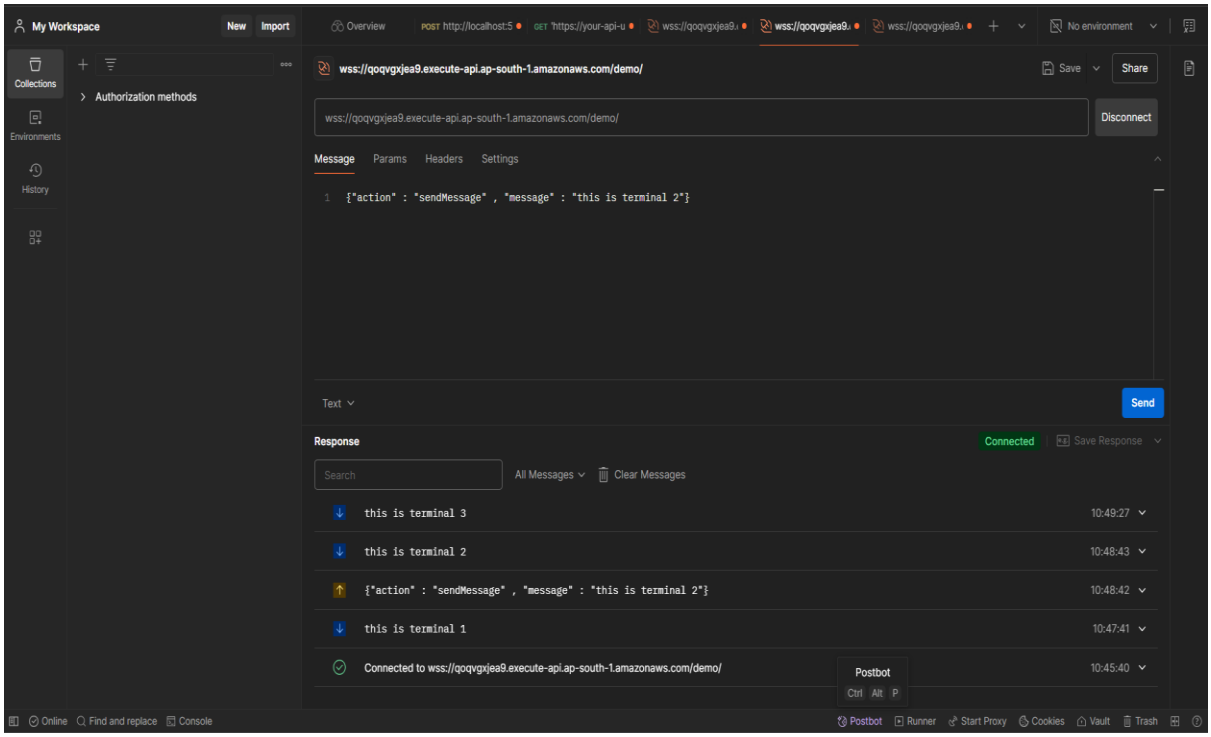


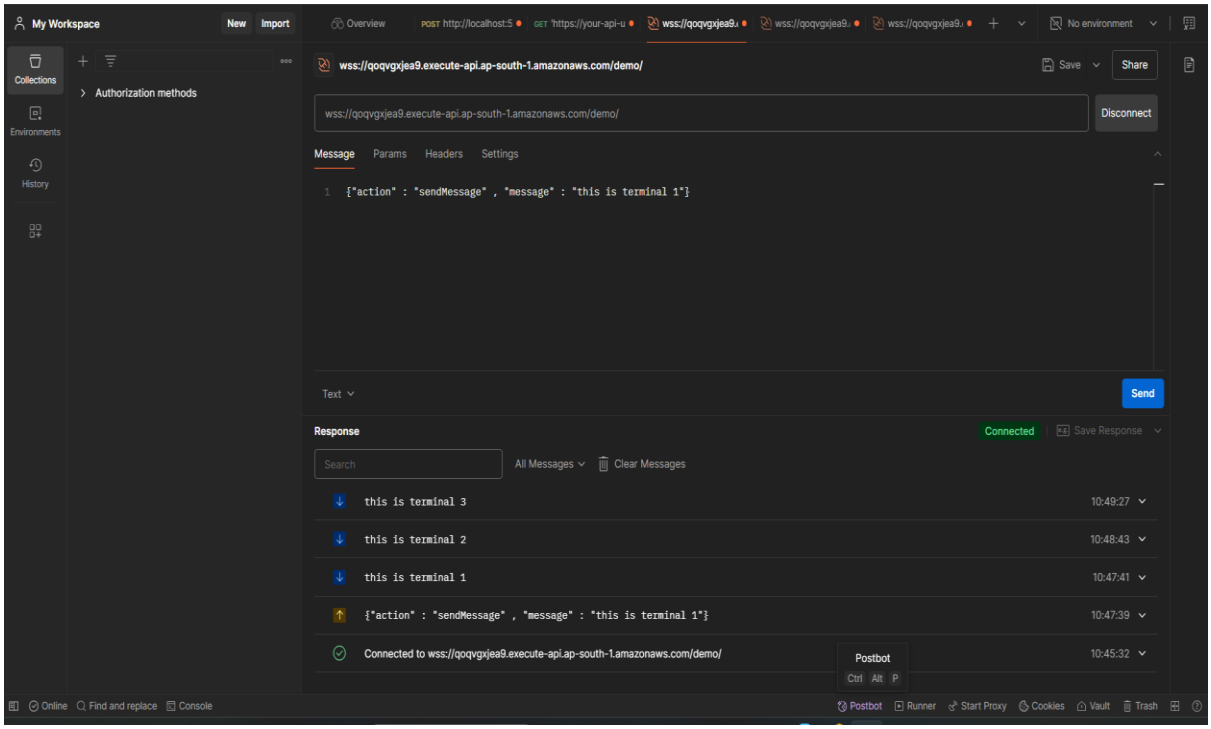



aws
Services
Search
[Alt+S]
Mumbai
rachanajs @ rjs-2000
CloudTrail
S3
Simple Notification Service
CloudWatch
EC2
IAM
VPC
EFS
Route 53
Certificate Manager
RDS
CloudFront
AWS Backup
Lambda
API Gateway
DynamoDB
The websocket-connections table was created successfully.
DynamoDB > Tables
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations New
Reserved capacity
Settings
DAX
Clusters
Subnet groups
Parameter groups
Events
Tables (1) Info
Actions
Delete
Create table
Find tables
Any tag key
Any tag value
1 match
Name
Status
Partition key
Sort key
Indexes
Deletion protection
Favorite
Read capacity m
websocket-connections
Active
connectionId (S)
-
0
Off
Provisioned (5)
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


aws
Services
Search
[Alt+S]
Mumbai
rachanajs @ rjs-2000
CloudTrail
S3
Simple Notification Service
CloudWatch
EC2
IAM
VPC
EFS
Route 53
Certificate Manager
RDS
CloudFront
AWS Backup
Lambda
API Gateway
API Gateway
APIs
Custom domain names
VPC links
API: websocket-api
Routes
Stages
Authorizers
Models
Dashboard
API settings
Usage plans
API keys
Client certificates
Settings
API Gateway > APIs > Routes - websocket-api (qoqvgxjea9)
Routes
Delete API
Deploy API
Create route
Route selection expression Info
$request.body.action
$connect
$disconnect
sendMessage
$connect
Enable two-way communication
Delete
ARN
arn:aws:execute-api:ap-south-1:711387118718:qoqvgxjea9/*/$connect
Client
Route request
Integration request
Lambda integration
Route request
Integration request
Integration response
Route response
Route request settings
Edit
Configure how clients will be authorized to access this WebSocket API. Use the $connect route to invoke an integration when a client attempts to connect to your API. You can specify authorization settings only for the $connect route.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


My Workspace
New
Import
Overview
POST http://localhost:5
GET https://your-api-u
wss://qoqvgxjea9.
wss://qoqvgxjea9.
wss://qoqvgxjea9.
No environment
Collections
Environments
History
Authorization methods
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Save
Share
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Disconnect
Message
Params
Headers
Settings
1 {"action" : "sendMessage" , "message" : "this is terminal 3"}
Text
Send
Response
Connected
Save Response
Search
All Messages
Clear Messages
this is terminal 3
10:49:27
{"action" : "sendMessage" , "message" : "this is terminal 3"}
10:49:27
this is terminal 2
10:48:43
this is terminal 1
10:47:41
Connected to wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
10:45:47
Postbot
Ctrl Alt P
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


My Workspace
New
Import
Overview
POST http://localhost:5
GET https://your-api-u
wss://qoqvgxjea9.
wss://qoqvgxjea9.
wss://qoqvgxjea9.
No environment
Collections
Environments
History
Authorization methods
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Save
Share
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Disconnect
Message
Params
Headers
Settings
1 {"action" : "sendMessage" , "message" : "this is terminal 2"}
Text
Send
Response
Connected
Save Response
Search
All Messages
Clear Messages
this is terminal 3
10:49:27
this is terminal 2
10:48:43
{"action" : "sendMessage" , "message" : "this is terminal 2"}
10:48:42
this is terminal 1
10:47:41
Connected to wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
10:45:40
Postbot
Ctrl Alt P
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


My Workspace
New
Import
Overview
POST http://localhost:5
GET https://your-api-u
wss://qoqvgxjea9.
wss://qoqvgxjea9.
wss://qoqvgxjea9.
No environment
Collections
Environments
History
Authorization methods
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Save
Share
wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
Disconnect
Message
Params
Headers
Settings
1 {"action" : "sendMessage" , "message" : "this is terminal 1"}
Text
Send
Response
Connected
Save Response
Search
All Messages
Clear Messages
this is terminal 3
10:49:27
this is terminal 2
10:48:43
this is terminal 1
10:47:41
{"action" : "sendMessage" , "message" : "this is terminal 1"}
10:47:39
Connected to wss://qoqvgxjea9.execute-api.ap-south-1.amazonaws.com/demo/
10:45:32
Postbot
Ctrl Alt P
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash