

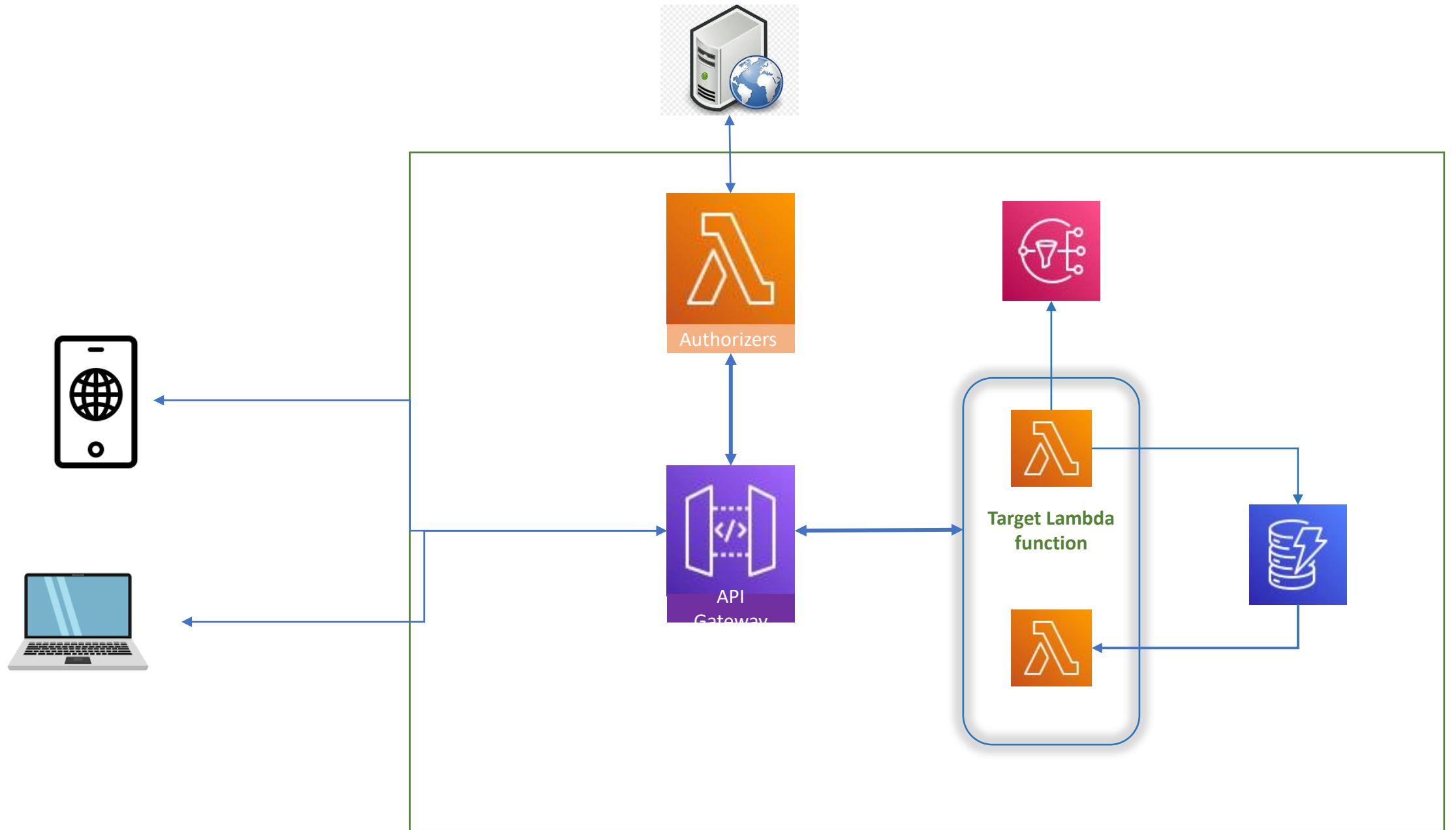
Send the notifications to user using
serverless framework using external
Authentication and Authorization

Sudhakar

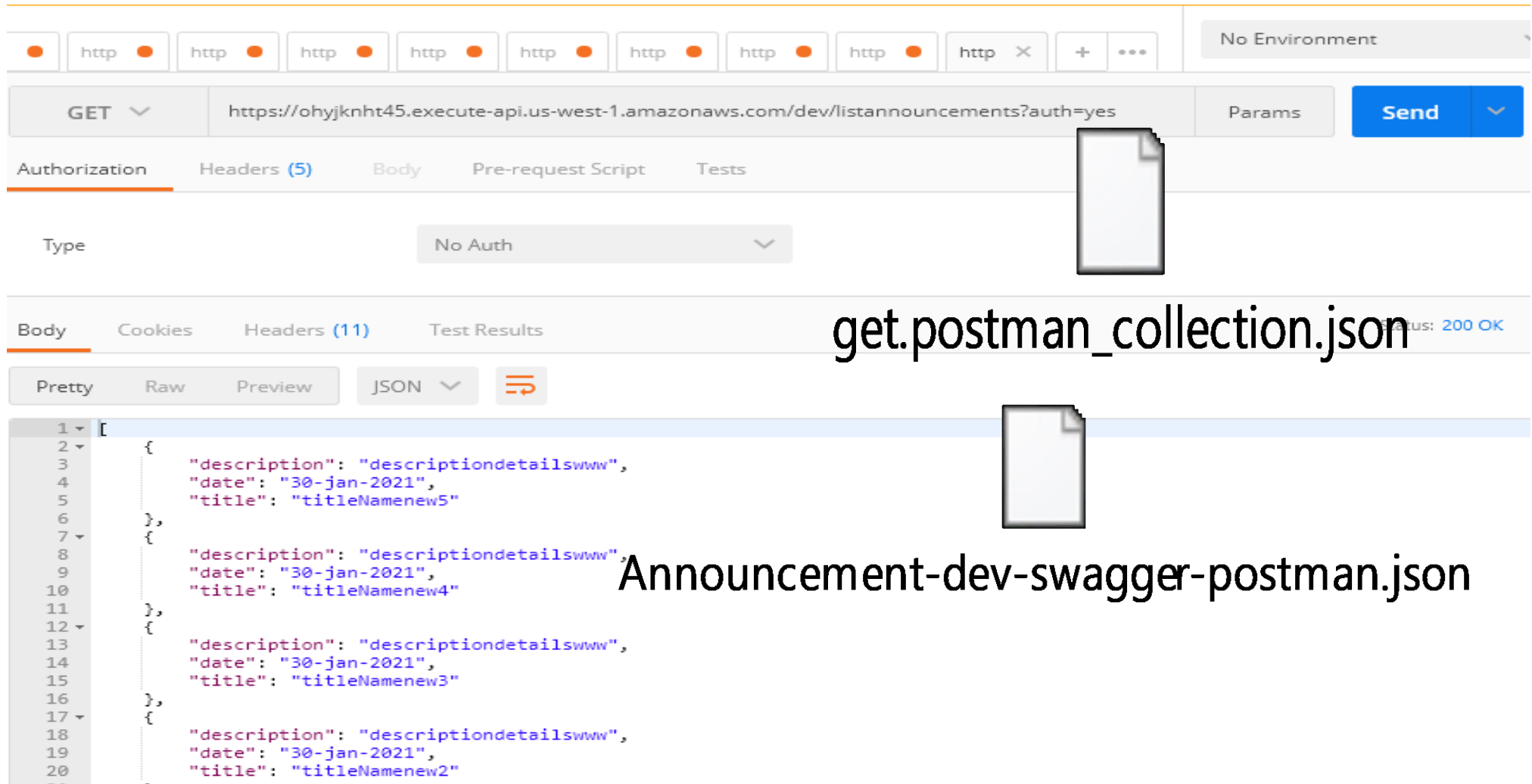
API Gateway Lambda Authorizers

You have an external Authentication system (Hosted as Authentication and Authorization server) that handles user authentication and issue tokens for authenticated users and this same system needs to be used for AWS API Gateway endpoint security as well. Meaning the token issued from the external Authentication System needs to be passed for each and every request to the API Gateway as well. This is where Lambda Authorizers come in. You can create a Lambda function that will be invoked every time a request is made to an API Gateway endpoint and you can write your own custom code to verify that token sent to the request is valid by sending the token to the external Authentication system and check for validity.

- Create an API Gateway.
- Create Lambda functions.
- Connect a Lambda function to an API Gateway.
- Add Authorizers to AWS API Gateway.
- Create SNS
- Create the DynamoDB table



GET /dev/listannouncements?auth=yes HTTP/1.1 Host: ohyjknht45.execute-api.us-west-1.amazonaws.com approval: approve Cache-Control: no-cache Postman-Token: c53a2b2e-5dde-2778-1ceb-a93814d769cf



The screenshot displays the Postman interface for a GET request. The URL bar shows the endpoint `https://ohyjknht45.execute-api.us-west-1.amazonaws.com/dev/listannouncements?auth=yes`. The 'Headers' tab is active, showing 11 headers. The 'Body' tab is selected, displaying a JSON array of four announcement objects. The status bar at the bottom right indicates a 200 OK response.

GET `https://ohyjknht45.execute-api.us-west-1.amazonaws.com/dev/listannouncements?auth=yes` Params Send

Authorization Headers (5) Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (11) Test Results Status: 200 OK

Pretty Raw Preview JSON

```
[
  {
    "description": "descriptiondetailswww",
    "date": "30-jan-2021",
    "title": "titleNamenew5"
  },
  {
    "description": "descriptiondetailswww",
    "date": "30-jan-2021",
    "title": "titleNamenew4"
  },
  {
    "description": "descriptiondetailswww",
    "date": "30-jan-2021",
    "title": "titleNamenew3"
  },
  {
    "description": "descriptiondetailswww",
    "date": "30-jan-2021",
    "title": "titleNamenew2"
  }
]
```

get.postman_collection.json

Announcement-dev-swagger-postman.json

POST /dev/addannouncements?auth=yes HTTP/1.1 Host: ohyjknht45.execute-api.us-west-1.amazonaws.com approval:
approveCache-Control: no-cache Postman-Token: 98dedfc0-1fb6-a0e3-c687-5f4e50c52f2c {"description":
"descriptiondetailswww", "date": "30-jan-2021", "title": "titleNamenew6"}

The image shows a Postman interface for a POST request. The top bar displays the method 'POST', the URL 'https://ohyjknht45.execute-api.us-west-1.amazonaws.com/dev/addannouncements?auth=yes', and a 'Send' button. Below the bar, tabs for 'Authorization', 'Headers (5)', 'Body', 'Pre-request Script', and 'Tests' are visible. The 'Body' tab is selected, showing a JSON payload:

```
1 {"description": "descriptiondetailswww", "date": "30-jan-2021", "title": "titleNamenew6"}
```

. Below the body, a large document icon is shown. The bottom section displays the response, with tabs for 'Body', 'Cookies', 'Headers (11)', and 'Test Results'. The 'Body' tab is selected, showing a JSON response:

```
1 {  
2   "message": "announcements entry created"  
3 }
```

. The status '200 OK' is visible on the right.

post.postman_collection.json