Unit Tests - Backend DataBase Writes

Written By: Kavan Samra Team: Share Yourself Artists

About the code

My code was written using AWS Lambda, which is a compute platform that allows developers to write anonymous, highly-scalable functions in the language of their choice.

All code relating to this test data can be found in our Github Repo: ddavisscott/sya-react-aws/backend

Methodology

The interesting thing about using Lambda to host your code is that it makes it very simple to invoke the function with test data that you build. Lambda then replies with a response object that determines if the code ran without error. In order to see if I correctly inserted into the DB the fields I expected, I then check the TEST Database for confirmation.

All of the lambda functions look like the following:

'''javascript

```
exports.handler = (event,context,callback) => {}
```

'''

Sample Test Data

event contains the actual data passed to the lambda function, this includes the test data that I pass in. **callback** allows me to return responses to the node console for debugging purposes. All test data is passed in as a JSON object.

Unit Tests

updateProfile.js

Goal: Updates the user whose update their respective entry in the database with new data.

Test Data:

'''

```
{
    "body": {
```

```
        "userID":"c2bcbc67-9699-4795-806a-0c703d408f1f",
        "role": "business",
        "instagram": "cityScapes,
        "facebook":,
        "twitter":,
        "tumblr":,
        "about":,
        "additionalNotes":,
        "worthKnowing":,
        "url":
    }
}
```

'''

Result: Success! userID: c2bcbc67-9699-4795-806a-0c703d408f1f has new field name "instagram" with the value "cityScapes"

sendReviewRequest.js

Goal: Allows artists to send a review request to businesses, along with the artwork they have chosen.

Test Data:

'''

```
{
    "body": {
        "artTitle": "Mona Lisa",
        "artistName": "TheKing",
        "artistID": "c2bcbc67-9699-4795-806a-0c703d408f1f",
        "artistEmail": "test@gmail.com",
        "artDescription": "THE Mona Lisa",
        "uploadDate": "Long Ago",
        "url": "image-url",
        "artistCredit": "10",
        "businessName": "NY Times",
        "businessEmail": "nytimes@ny.com",
        "businessID": "some uuid",
        "submittedWithFreeCredit": "true"
    }
}
```

'''

Result: "Success! The specified data was placed into the reviewTable."

Test Data 2: This time, since the user making the request has 0 credits, the request should *fail*

'''

```
{
    "body": {
        "artTitle": "Mona Lisa",
        "artistName": "TheKing",
        "artistID": "c2bcbc67-9699-4795-806a-0c703d408f1f",
        "artistEmail": "test@gmail.com",
        "artDescription": "THE Mona Lisa",
        "uploadDate": "Long Ago",
        "url": "image-url",
        "artistCredit": "0",
        "businessName": "NY Times",
        "businessEmail": "nytimes@ny.com",
        "businessID": "some uuid",
        "submittedWithFreeCredit": "true"
    }
}
```

'''

Result: "Failed! The user did not have enough credits to submit a request"

sendReviewResponse.js

Goal: Allow businesses to respond to a review request"

Test Data:

'''

```
{
    "body": {
        "artTitle": "Mona Lisa",
        "artistName": "TheKing",
        "artistID": "c2bcbc67-9699-4795-806a-0c703d408f1f",
        "artistEmail": "test@gmail.com",
        "artDescription": "THE Mona Lisa",
        "uploadDate": "Long Ago",
        "url": "image-url",
        "artistCredit": "10",
        "businessName": "NY Times",
        "businessEmail": "nytimes@ny.com",
        "businessID": "some uuid",
        "submittedWithFreeCredit": "true",

        "reply": "This is a reply",
        "radios": "denied",
        "repliedDate": "today",
        "replied": "true"
    }
}
```

'''

Result: Success! The reviewRequest table has been updated with the new fields: "reply, radios, repliedDate, replied"

writeArtMetaDataToDB.js

Goal: When an image is placed into the S3 bucket, upload the relative art metadata to the artwork database.

Test Data:

'''

```
{
    "body": {
        "imageName": "monaLisa",
        "artistName": "Leo Davinci",
        "artistID": "an ID",
        "artDescription": "this is my masterpiece",
        "uploadDate": "long ago",
        "sourceKey": "monalisa"
    }
}
```

'''

Result: Success! The data was placed in the artwork table

fillUserTable.js

Goal: Write user data to user table after a user is confirmed to have signed up.

Test Data:

'''

```
{
    "body": {
        "userID": "someID",
        "email": "test@gmail.com",
        "dateCreated": "today"
        "username": "tomorrow"
        "role": "artist"
    }
}
```

'''

Result: Success! The information has been placed into the user database.