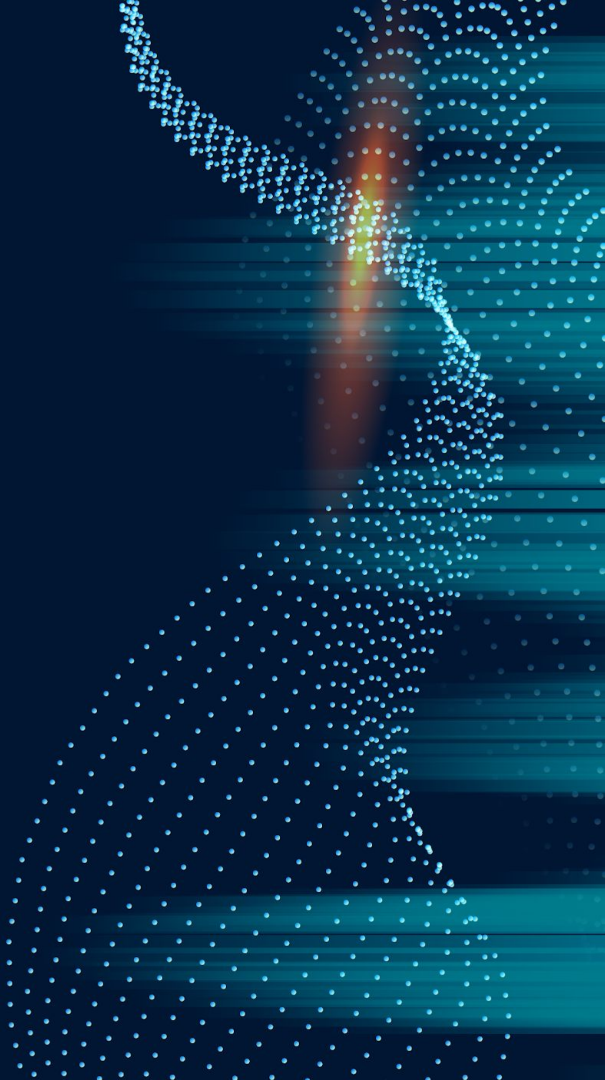
Swaous
Risen One Project 1

Joshua Jackson, Dakota Reid, Jakob Bush, Jackson Boster

Goal of Project 1 Review

Implement a section of the Risen One Consulting Employee Portal, where employees can submit daily status reports.

There is to be daily reporting documentation of each employee, where they can create daily, monthly, and yearly reports



Team DIstribution

Front End Development

Dakota Reid

Jakob Bush

Back End Development

Joshua Jackson

Jackson Boster

Goals and Learning

Frontend

Angular - FAQS

Learning Examples in Blackboard

Backend

AWS - FAQs

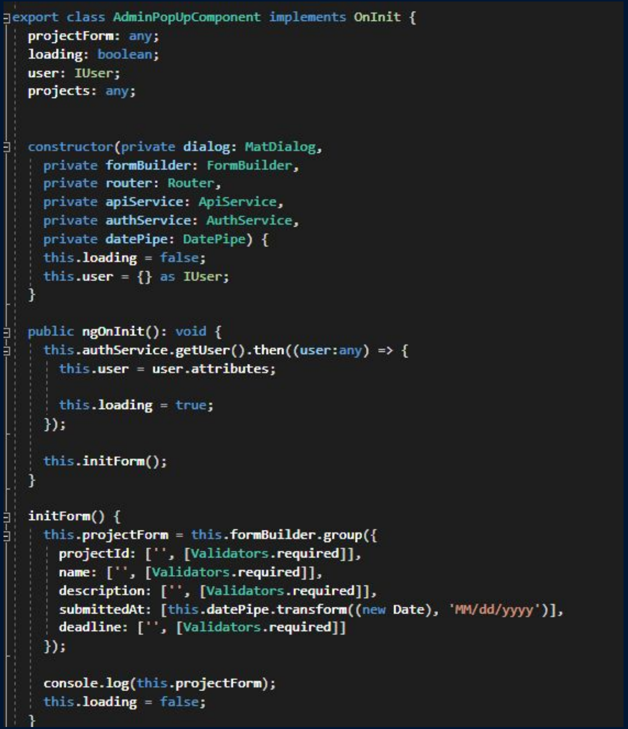
Udemy

DevOps CoWorkers

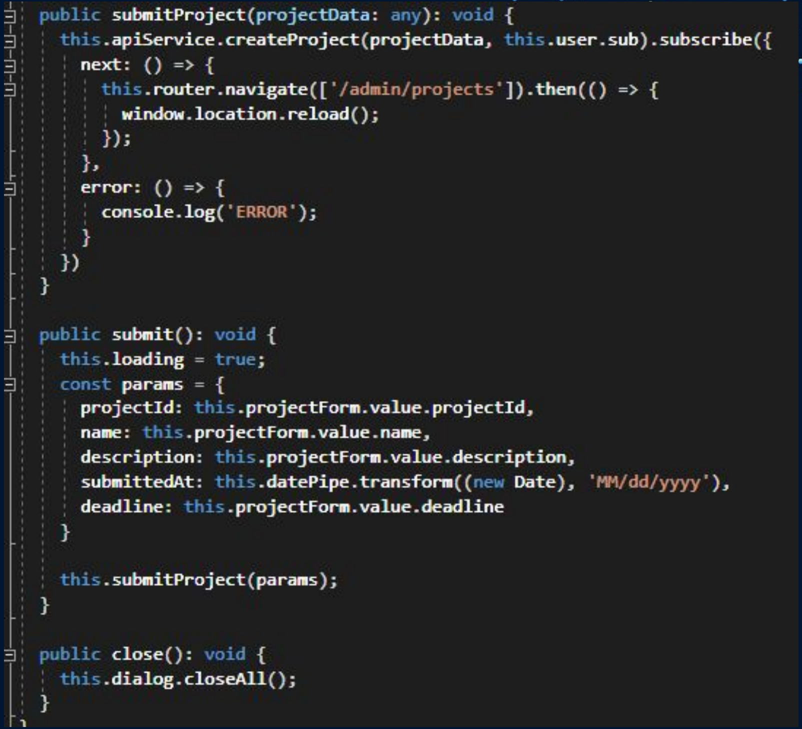
Demonstration

RISEN ONE CONSULTING

Front End Explanation: Admin Component

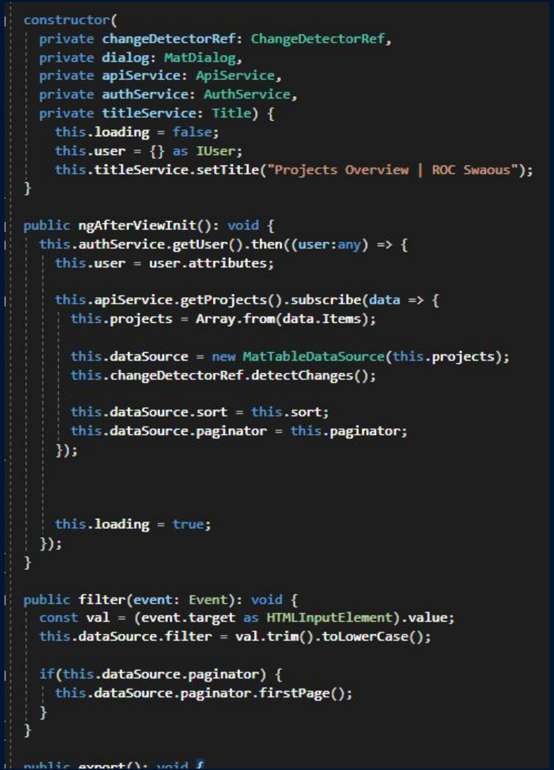


```
export class AdminPopUpComponent implements OnInit {
  projectForm: any;
  loading: boolean;
  user: IUser;
  projects: any;


  constructor(private dialog: MatDialog,
    private formBuilder: FormBuilder,
    private router: Router,
    private apiService: ApiService,
    private authService: AuthService,
    private datePipe: DatePipe) {
    this.loading = false;
    this.user = {} as IUser;
  }

  public ngOnInit(): void {
    this.authService.getUser().then((user:any) => {
      this.user = user.attributes;

      this.loading = true;
    });

    this.initForm();
  }

  initForm() {
    this.projectForm = this.formBuilder.group({
      projectId: ['', [Validators.required]],
      name: ['', [Validators.required]],
      description: ['', [Validators.required]],
      submittedAt: [this.datePipe.transform((new Date), 'MM/dd/yyyy')],
      deadline: ['', [Validators.required]]
    });

    console.log(this.projectForm);
    this.loading = false;
  }
```

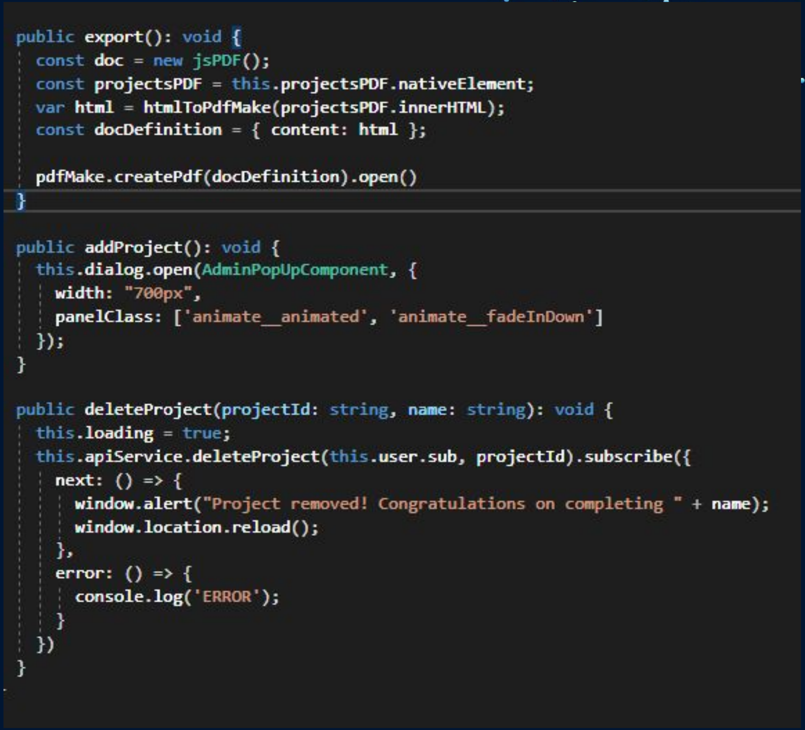


```
  public submitProject(projectData: any): void {
    this.apiService.createProject(projectData, this.user.sub).subscribe({
      next: () => {
        this.router.navigate(['/admin/projects']).then(() => {
          window.location.reload();
        });
      },
      error: () => {
        console.log('ERROR');
      }
    })
  }

  public submit(): void {
    this.loading = true;
    const params = {
      projectId: this.projectForm.value.projectId,
      name: this.projectForm.value.name,
      description: this.projectForm.value.description,
      submittedAt: this.datePipe.transform((new Date), 'MM/dd/yyyy'),
      deadline: this.projectForm.value.deadline
    }

    this.submitProject(params);
  }

  public close(): void {
    this.dialog.closeAll();
  }
}
```

Front End Explanation: Admin Projects

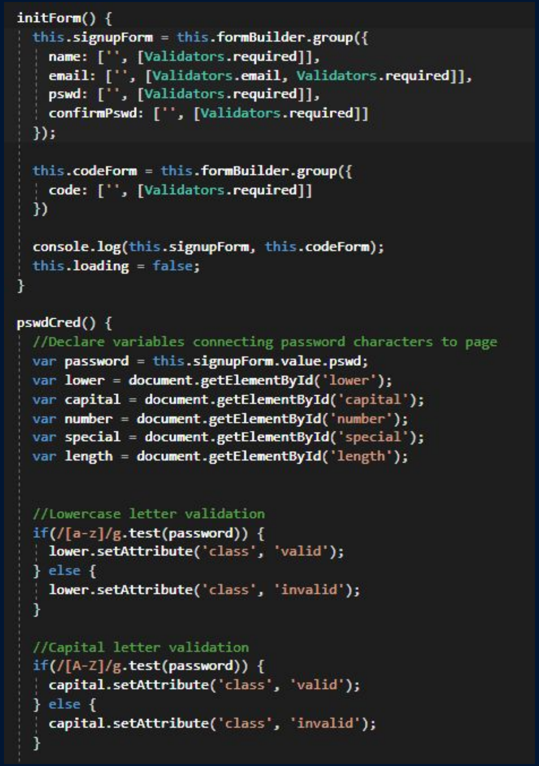


```
constructor(
  private changeDetectorRef: ChangeDetectorRef,
  private dialog: MatDialog,
  private apiService: ApiService,
  private authService: AuthService,
  private titleService: Title) {
    this.loading = false;
    this.user = {} as IUser;
    this.titleService.setTitle("Projects Overview | ROC Swaous");
}

public ngAfterViewInit(): void {
  this.authService.getUser().then((user:any) => {
    this.user = user.attributes;

    this.apiService.getProjects().subscribe(data => {
      this.projects = Array.from(data.Items);

      this.dataSource = new MatTableDataSource(this.projects);
      this.changeDetectorRef.detectChanges();

      this.dataSource.sort = this.sort;
      this.dataSource.paginator = this.paginator;
    });


    this.loading = true;
  });
}

public filter(event: Event): void {
  const val = (event.target as HTMLInputElement).value;
  this.dataSource.filter = val.trim().toLowerCase();

  if(this.dataSource.paginator) {
    this.dataSource.paginator.firstPage();
  }
}

public export(): void {
```

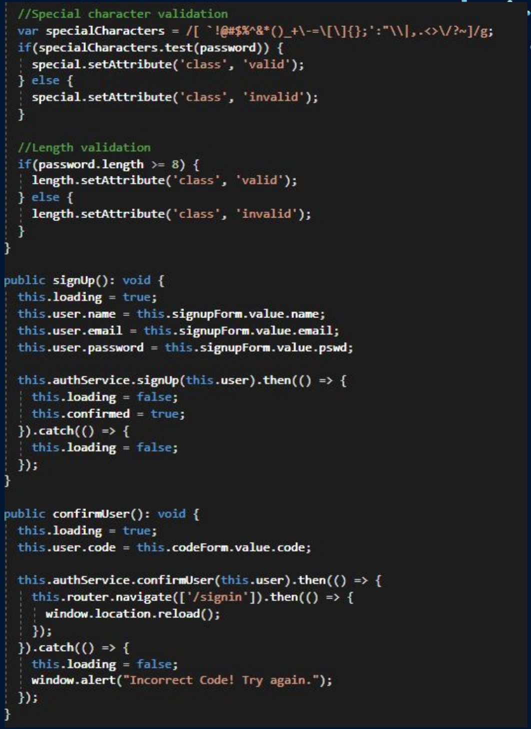


```
public export(): void {
  const doc = new jsPDF();
  const projectsPDF = this.projectsPDF.nativeElement;
  var html = htmlToPdfMake(projectsPDF.innerHTML);
  const docDefinition = { content: html };

  pdfMake.createPdf(docDefinition).open()
}

public addProject(): void {
  this.dialog.open(AdminPopUpComponent, {
    width: "700px",
    panelClass: ['animate__animated', 'animate__fadeInDown']
  });
}

public deleteProject(projectId: string, name: string): void {
  this.loading = true;
  this.apiService.deleteProject(this.user.sub, projectId).subscribe({
    next: () => {
      window.alert("Project removed! Congratulations on completing " + name);
      window.location.reload();
    },
    error: () => {
      console.log('ERROR');
    }
  })
}
```

Front End Explanation: Sign Up

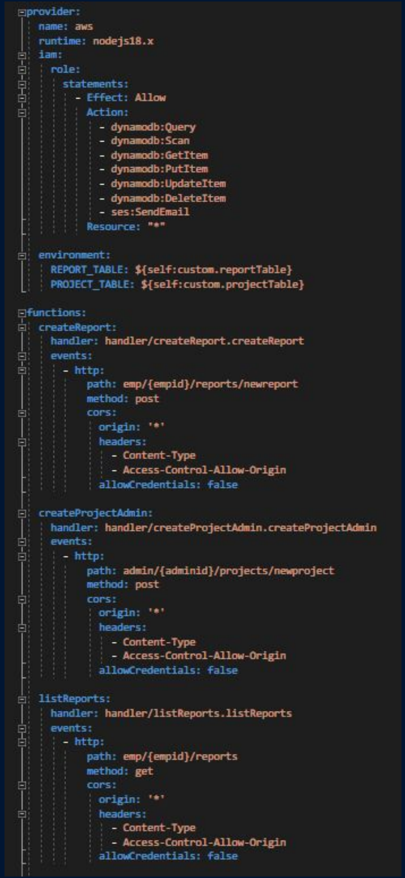


```
initForm() {
  this.signupForm = this.formBuilder.group({
    name: ['', [Validators.required]],
    email: ['', [Validators.email, Validators.required]],
    pswd: ['', [Validators.required]],
    confirmPswd: ['', [Validators.required]]
  });

  this.codeForm = this.formBuilder.group({
    code: ['', [Validators.required]]
  })

  console.log(this.signupForm, this.codeForm);
  this.loading = false;
}

pswdCred() {
  //Declare variables connecting password characters to page
  var password = this.signupForm.value.pswd;
  var lower = document.getElementById('lower');
  var capital = document.getElementById('capital');
  var number = document.getElementById('number');
  var special = document.getElementById('special');
  var length = document.getElementById('length');


  //Lowercase letter validation
  if(/[a-z]/g.test(password)) {
    lower.setAttribute('class', 'valid');
  } else {
    lower.setAttribute('class', 'invalid');
  }

  //Capital letter validation
  if(/[A-Z]/g.test(password)) {
    capital.setAttribute('class', 'valid');
  } else {
    capital.setAttribute('class', 'invalid');
  }
```

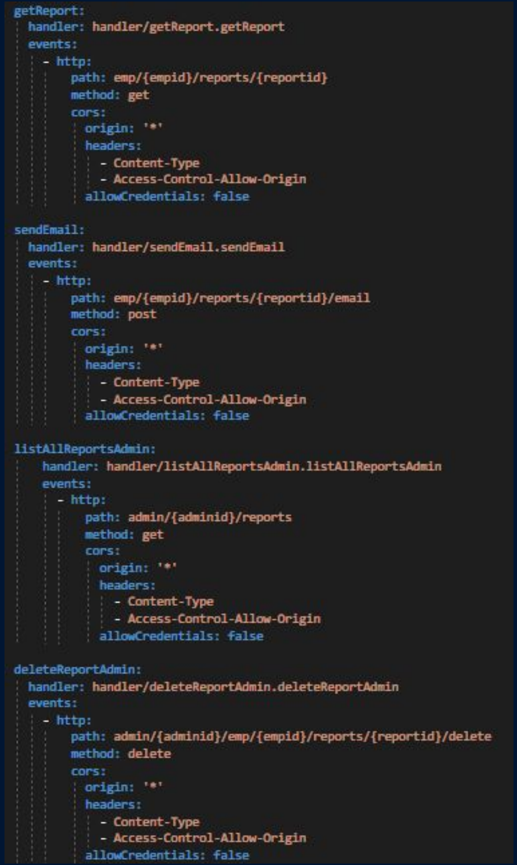


```
  //Special character validation
  var specialCharacters = /[ `!@#$%^&*()_+\-=\[\]{};':"\\|,.<>\/?~]/g;
  if(specialCharacters.test(password)) {
    special.setAttribute('class', 'valid');
  } else {
    special.setAttribute('class', 'invalid');
  }

  //Length validation
  if(password.length >= 8) {
    length.setAttribute('class', 'valid');
  } else {
    length.setAttribute('class', 'invalid');
  }
}

public signUp(): void {
  this.loading = true;
  this.user.name = this.signupForm.value.name;
  this.user.email = this.signupForm.value.email;
  this.user.password = this.signupForm.value.pswd;

  this.authService.signUp(this.user).then(() => {
    this.loading = false;
    this.confirmed = true;
  }).catch(() => {
    this.loading = false;
  });
}

public confirmUser(): void {
  this.loading = true;
  this.user.code = this.codeForm.value.code;

  this.authService.confirmUser(this.user).then(() => {
    this.router.navigate(['/signin']).then(() => {
      window.location.reload();
    });
  }).catch(() => {
    this.loading = false;
    window.alert("Incorrect Code! Try again.");
  });
}
```

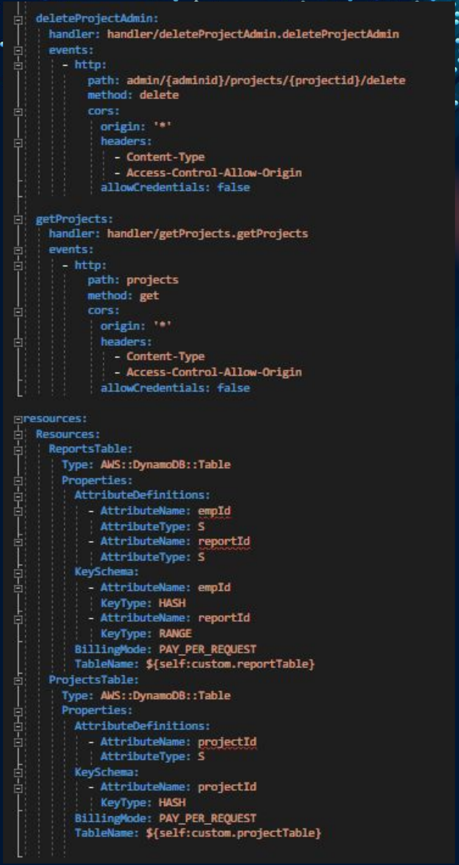
Back End Explanation: Serverless YAML



```yaml
provider:
  name: aws
  runtime: nodejs18.x
  iam:
    role:
      statements:
        - Effect: Allow
          Action:
            - dynamodb:Query
            - dynamodb:Scan
            - dynamodb:GetItem
            - dynamodb:PutItem
            - dynamodb:UpdateItem
            - dynamodb:DeleteItem
            - ses:SendEmail
          Resource: "*"

  environment:
    REPORT_TABLE: ${self:custom.reportTable}
    PROJECT_TABLE: ${self:custom.projectTable}

functions:
  createReport:
    handler: handler/createReport.createReport
    events:
      - http:
          path: emp/{empid}/reports/newreport
          method: post
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false

  createProjectAdmin:
    handler: handler/createProjectAdmin.createProjectAdmin
    events:
      - http:
          path: admin/{adminid}/projects/newproject
          method: post
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false

  listReports:
    handler: handler/listReports.listReports
    events:
      - http:
          path: emp/{empid}/reports
          method: get
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false
```

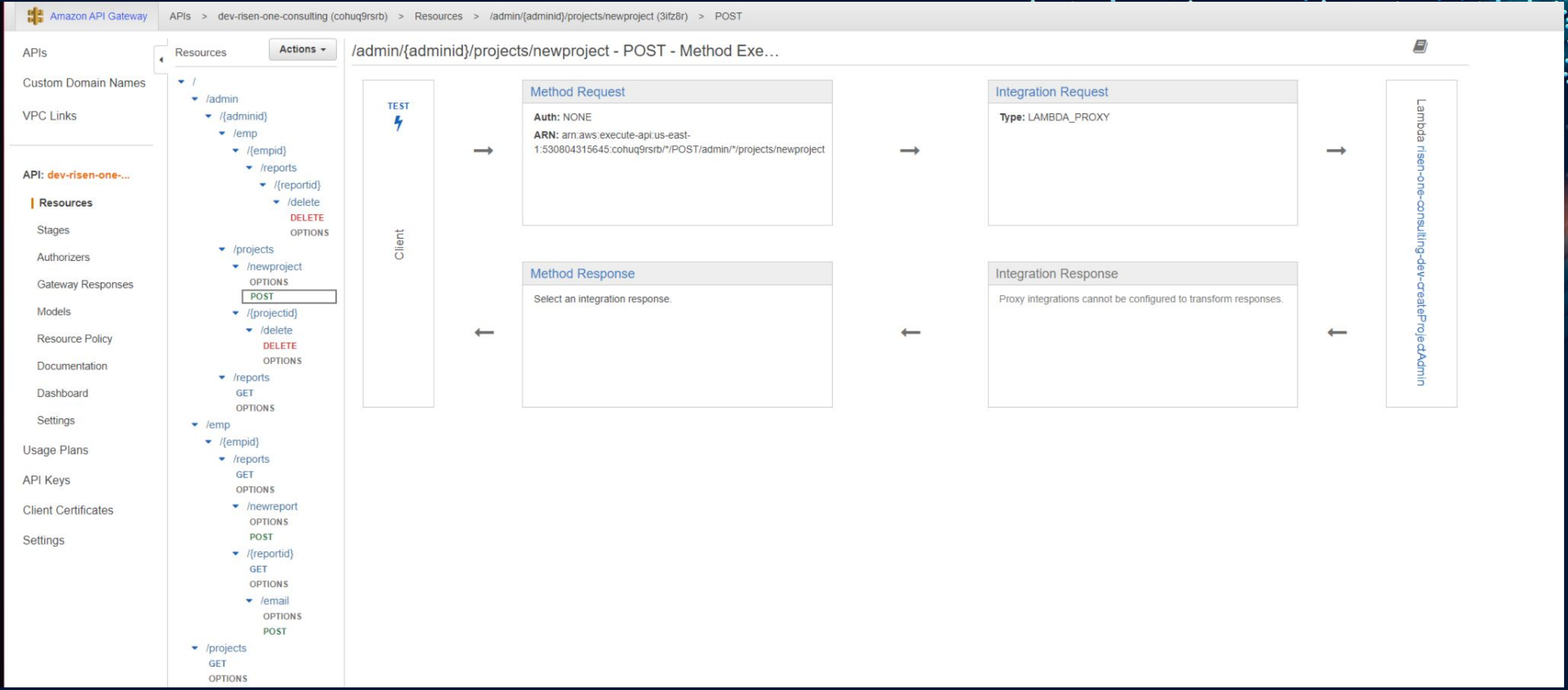


```yaml
getReport:
  handler: handler/getReport.getReport
  events:
    - http:
        path: emp/{empid}/reports/{reportid}
        method: get
        cors:
          origin: '*'
          headers:
            - Content-Type
            - Access-Control-Allow-Origin
          allowCredentials: false

sendEmail:
  handler: handler/sendEmail.sendEmail
  events:
    - http:
        path: emp/{empid}/reports/{reportid}/email
        method: post
        cors:
          origin: '*'
          headers:
            - Content-Type
            - Access-Control-Allow-Origin
          allowCredentials: false

listAllReportsAdmin:
    handler: handler/listAllReportsAdmin.listAllReportsAdmin
    events:
      - http:
          path: admin/{adminid}/reports
          method: get
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false

deleteReportAdmin:
  handler: handler/deleteReportAdmin.deleteReportAdmin
  events:
    - http:
        path: admin/{adminid}/emp/{empid}/reports/{reportid}/delete
        method: delete
        cors:
          origin: '*'
          headers:
            - Content-Type
            - Access-Control-Allow-Origin
          allowCredentials: false
```



```yaml
  deleteProjectAdmin:
    handler: handler/deleteProjectAdmin.deleteProjectAdmin
    events:
      - http:
          path: admin/{adminid}/projects/{projectid}/delete
          method: delete
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false

  getProjects:
    handler: handler/getProjects.getProjects
    events:
      - http:
          path: projects
          method: get
          cors:
            origin: '*'
            headers:
              - Content-Type
              - Access-Control-Allow-Origin
            allowCredentials: false

resources:
  Resources:
    ReportsTable:
      Type: AWS::DynamoDB::Table
      Properties:
        AttributeDefinitions:
          - AttributeName: empId
            AttributeType: S
          - AttributeName: reportId
            AttributeType: S
        KeySchema:
          - AttributeName: empId
            KeyType: HASH
          - AttributeName: reportId
            KeyType: RANGE
        BillingMode: PAY_PER_REQUEST
        TableName: ${self:custom.reportTable}
    ProjectsTable:
      Type: AWS::DynamoDB::Table
      Properties:
        AttributeDefinitions:
          - AttributeName: projectId
            AttributeType: S
        KeySchema:
          - AttributeName: projectId
            KeyType: HASH
        BillingMode: PAY_PER_REQUEST
        TableName: ${self:custom.projectTable}
```

Back End Explanation: API Gateway



Amazon API Gateway APIs > dev-risen-one-consulting (cohuq9rsrb) > Resources > /admin/{adminid}/projects/newproject (3ifz8r) > POST

APIs
Custom Domain Names
VPC Links

API: dev-risen-one-...
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Dashboard
Settings
Usage Plans
API Keys
Client Certificates
Settings

Resources Actions

```
/
  /admin
    /{adminid}
      /emp
        /{empid}
          /reports
            /{reportid}
              /delete
                DELETE
                OPTIONS
      /projects
        /newproject
          OPTIONS
          POST
        /{projectid}
          /delete
            DELETE
            OPTIONS
      /reports
        GET
        OPTIONS
  /emp
    /{empid}
      /reports
        GET
        OPTIONS
        /newreport
          OPTIONS
          POST
        /{reportid}
          GET
          OPTIONS
          /email
            OPTIONS
            POST
  /projects
    GET
    OPTIONS
```

/admin/{adminid}/projects/newproject - POST - Method Exe...

TEST

Client

Method Request

Auth: NONE

ARN: arn:aws:execute-api:us-east-1:530804315645:cohuq9rsrb/*/POST/admin/*/projects/newproject

Integration Request

Type: LAMBDA_PROXY

Lambda risen-one-consulting-dev-createProjectAdmin

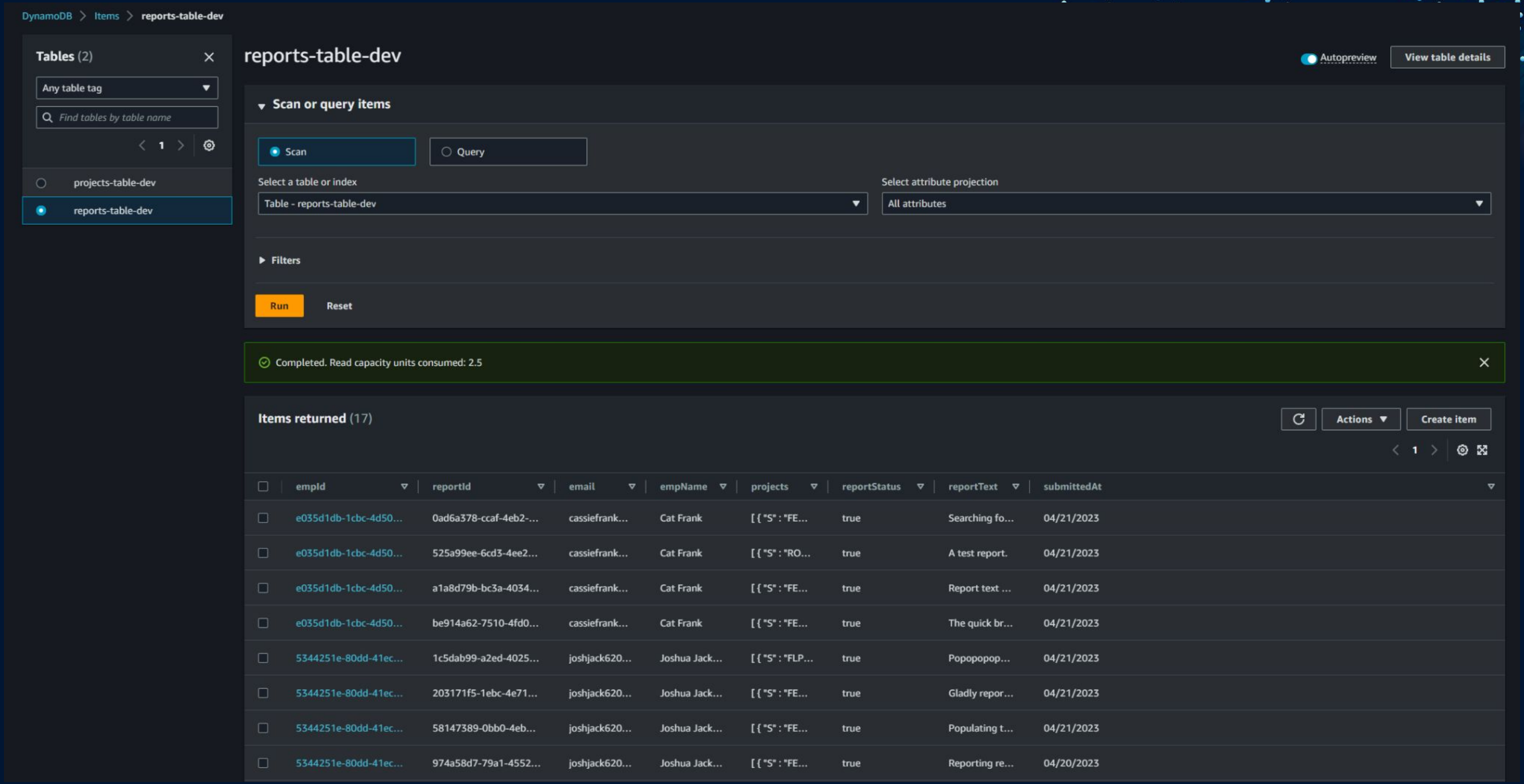
Method Response

Select an integration response.

Integration Response

Proxy integrations cannot be configured to transform responses.

Back End Explanation: Reports Table



reports-table-dev

Select a table or index: Table - reports-table-dev

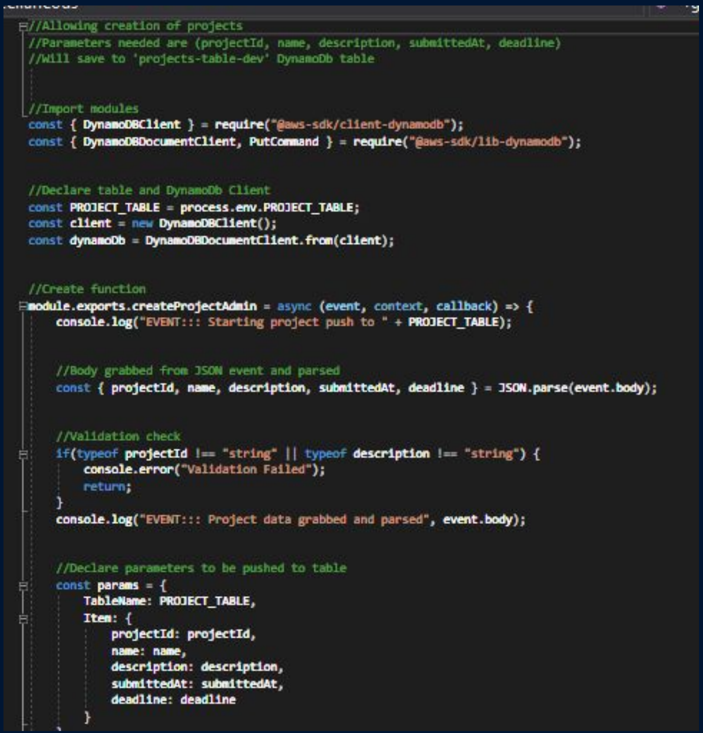
Select attribute projection: All attributes

Completed. Read capacity units consumed: 2.5

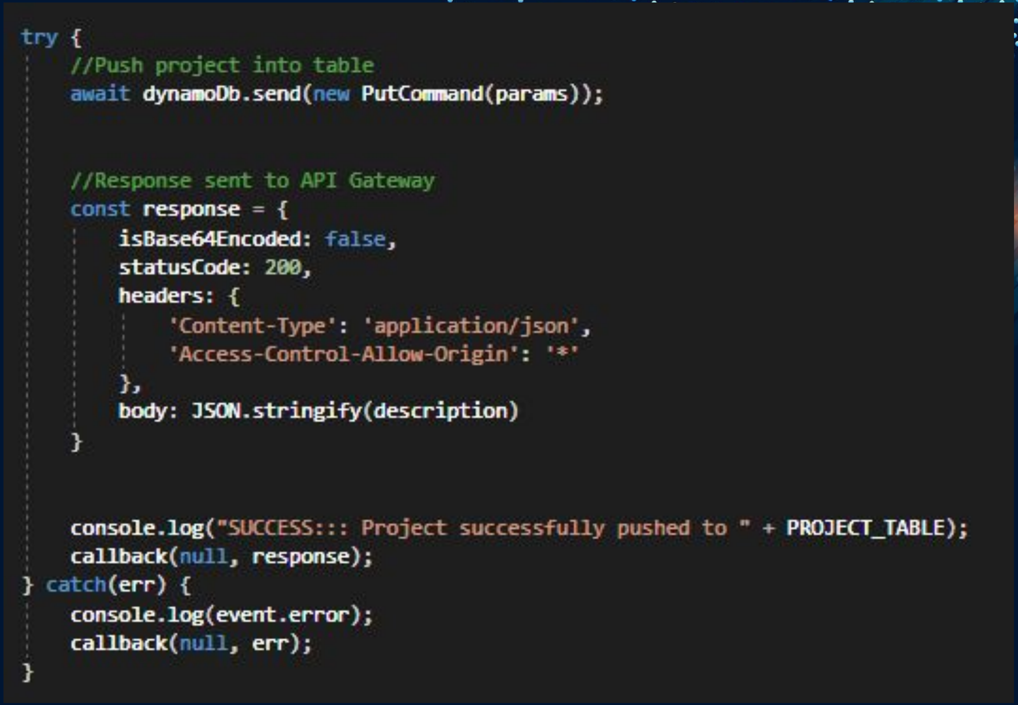
Items returned (17)

empId	reportId	email	empName	projects	reportStatus	reportText	submittedAt
e035d1db-1cbc-4d50...	0ad6a378-ccaf-4eb2-...	cassiefrank...	Cat Frank	[{ "S" : "FE...	true	Searching fo...	04/21/2023
e035d1db-1cbc-4d50...	525a99ee-6cd3-4ee2...	cassiefrank...	Cat Frank	[{ "S" : "RO...	true	A test report.	04/21/2023
e035d1db-1cbc-4d50...	a1a8d79b-bc3a-4034...	cassiefrank...	Cat Frank	[{ "S" : "FE...	true	Report text ...	04/21/2023
e035d1db-1cbc-4d50...	be914a62-7510-4fd0...	cassiefrank...	Cat Frank	[{ "S" : "FE...	true	The quick br...	04/21/2023
5344251e-80dd-41ec...	1c5dab99-a2ed-4025...	joshjack620...	Joshua Jack...	[{ "S" : "FLP...	true	Popopopop...	04/21/2023
5344251e-80dd-41ec...	203171f5-1ebc-4e71...	joshjack620...	Joshua Jack...	[{ "S" : "FE...	true	Gladly repor...	04/21/2023
5344251e-80dd-41ec...	58147389-0bb0-4eb...	joshjack620...	Joshua Jack...	[{ "S" : "FE...	true	Populating t...	04/21/2023
5344251e-80dd-41ec...	974a58d7-79a1-4552...	joshjack620...	Joshua Jack...	[{ "S" : "FE...	true	Reporting re...	04/20/2023

Back End Explanation: Admin Create Project

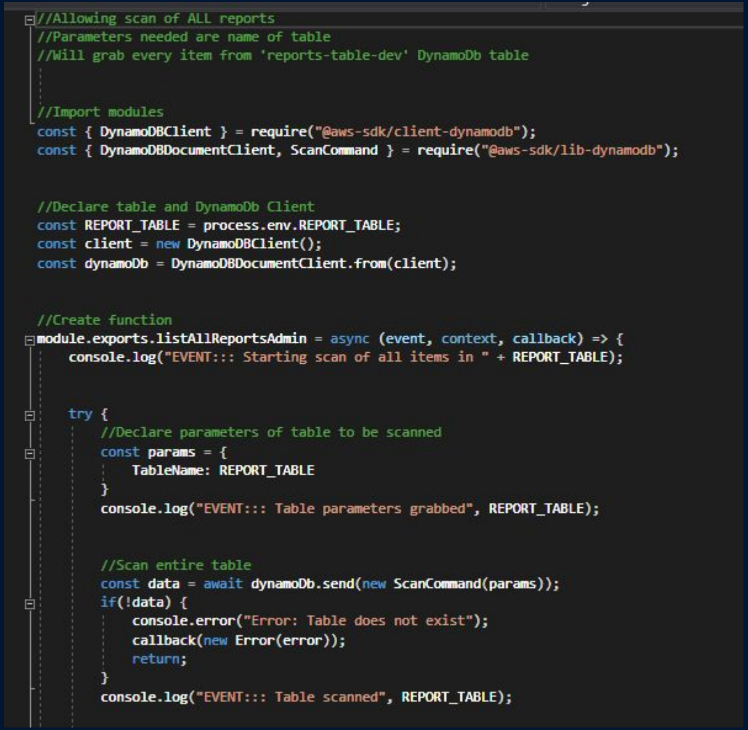


```javascript
//Allowing creation of projects
//Parameters needed are (projectId, name, description, submittedAt, deadline)
//Will save to 'projects-table-dev' DynamoDb table


//Import modules
const { DynamoDBClient } = require("@aws-sdk/client-dynamodb");
const { DynamoDBDocumentClient, PutCommand } = require("@aws-sdk/lib-dynamodb");


//Declare table and DynamoDb Client
const PROJECT_TABLE = process.env.PROJECT_TABLE;
const client = new DynamoDBClient();
const dynamoDb = DynamoDBDocumentClient.from(client);


//Create function
module.exports.createProjectAdmin = async (event, context, callback) => {
    console.log("EVENT::: Starting project push to " + PROJECT_TABLE);


    //Body grabbed from JSON event and parsed
    const { projectId, name, description, submittedAt, deadline } = JSON.parse(event.body);


    //Validation check
    if(typeof projectId !== "string" || typeof description !== "string") {
        console.error("Validation Failed");
        return;
    }
    console.log("EVENT::: Project data grabbed and parsed", event.body);


    //Declare parameters to be pushed to table
    const params = {
        TableName: PROJECT_TABLE,
        Item: {
            projectId: projectId,
            name: name,
            description: description,
            submittedAt: submittedAt,
            deadline: deadline
        }
```

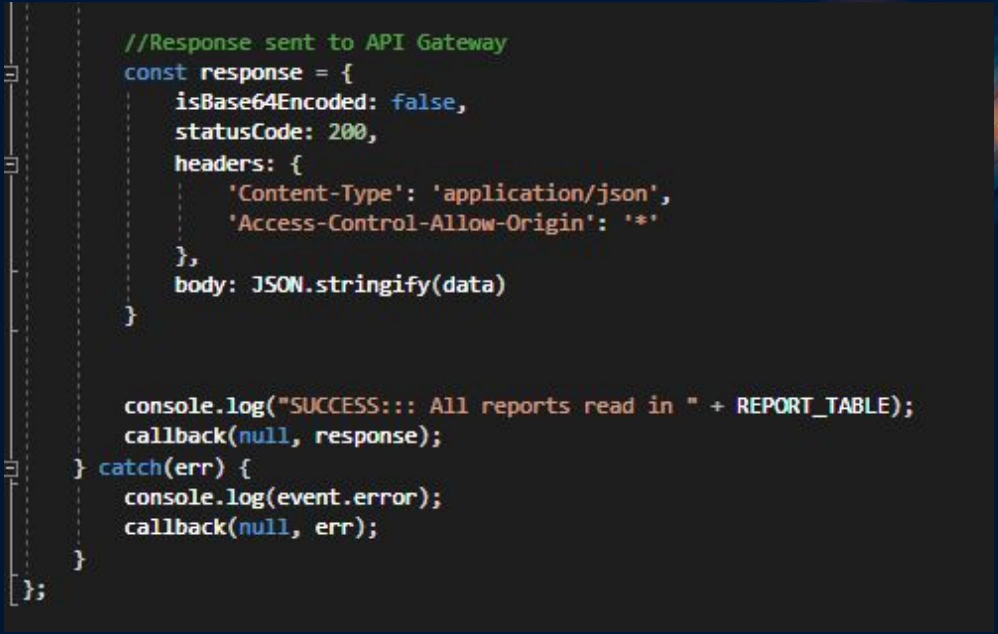


```javascript
try {
    //Push project into table
    await dynamoDb.send(new PutCommand(params));


    //Response sent to API Gateway
    const response = {
        isBase64Encoded: false,
        statusCode: 200,
        headers: {
            'Content-Type': 'application/json',
            'Access-Control-Allow-Origin': '*'
        },
        body: JSON.stringify(description)
    }


    console.log("SUCCESS::: Project successfully pushed to " + PROJECT_TABLE);
    callback(null, response);
} catch(err) {
    console.log(event.error);
    callback(null, err);
}
```

Back End Explanation: Grab All Reports

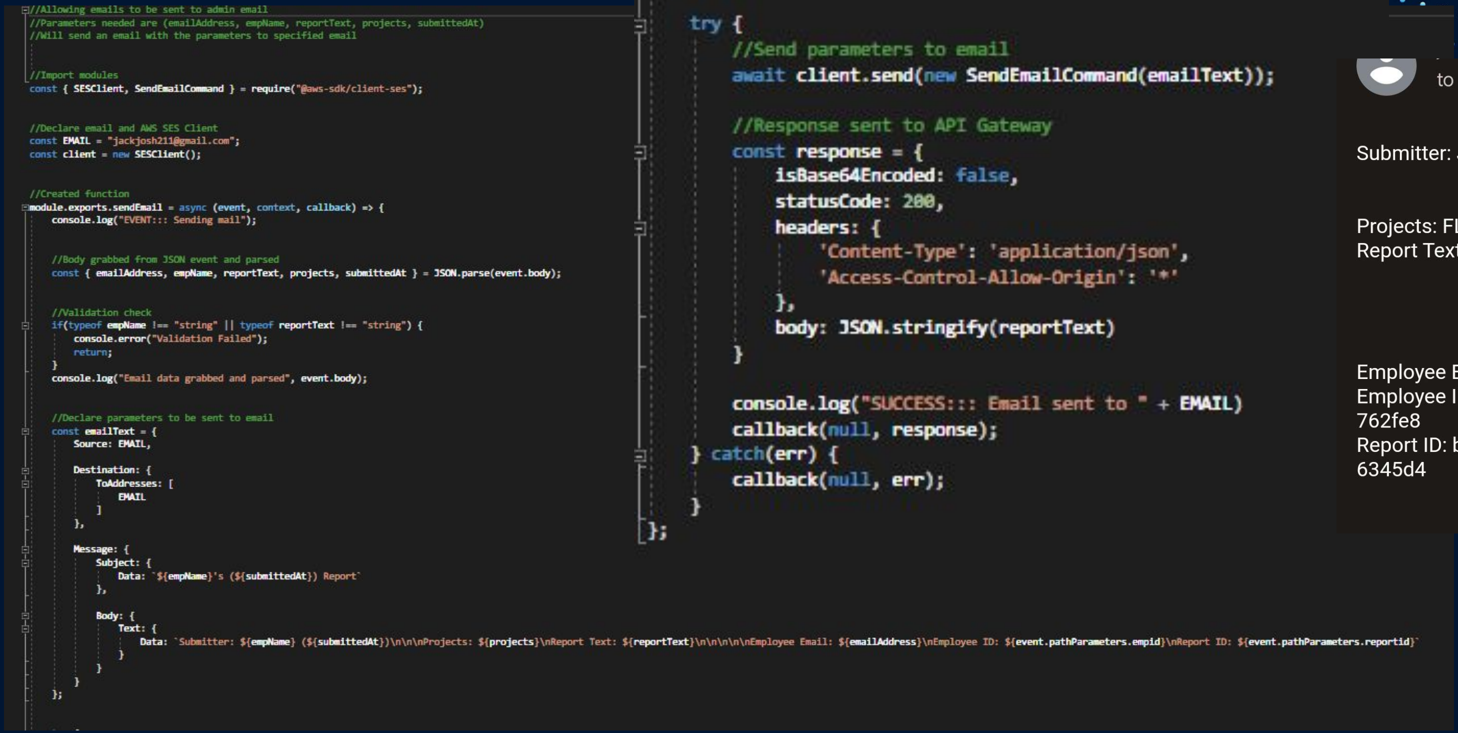


```
//Allowing scan of ALL reports
//Parameters needed are name of table
//Will grab every item from 'reports-table-dev' DynamoDb table


//Import modules
const { DynamoDBClient } = require("@aws-sdk/client-dynamodb");
const { DynamoDBDocumentClient, ScanCommand } = require("@aws-sdk/lib-dynamodb");


//Declare table and DynamoDb Client
const REPORT_TABLE = process.env.REPORT_TABLE;
const client = new DynamoDBClient();
const dynamoDb = DynamoDBDocumentClient.from(client);


//Create function
module.exports.listAllReportsAdmin = async (event, context, callback) => {
    console.log("EVENT::: Starting scan of all items in " + REPORT_TABLE);


    try {
        //Declare parameters of table to be scanned
        const params = {
            TableName: REPORT_TABLE
        }
        console.log("EVENT::: Table parameters grabbed", REPORT_TABLE);


        //Scan entire table
        const data = await dynamoDb.send(new ScanCommand(params));
        if(!data) {
            console.error("Error: Table does not exist");
            callback(new Error(error));
            return;
        }
        console.log("EVENT::: Table scanned", REPORT_TABLE);
```



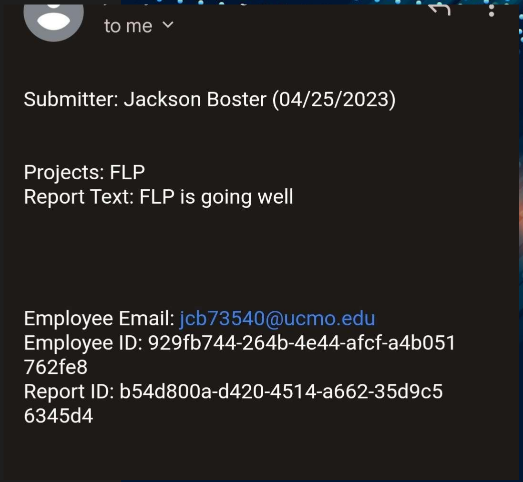
```
        //Response sent to API Gateway
        const response = {
            isBase64Encoded: false,
            statusCode: 200,
            headers: {
                'Content-Type': 'application/json',
                'Access-Control-Allow-Origin': '*'
            },
            body: JSON.stringify(data)
        }


        console.log("SUCCESS::: All reports read in " + REPORT_TABLE);
        callback(null, response);
    } catch(err) {
        console.log(event.error);
        callback(null, err);
    }
};
```

Back End Explanation: Project -> Email



```
//Allowing emails to be sent to admin email
//Parameters needed are (emailAddress, empName, reportText, projects, submittedAt)
//Will send an email with the parameters to specified email


//Import modules
const { SESClient, SendEmailCommand } = require("@aws-sdk/client-ses");


//Declare email and AWS SES Client
const EMAIL = "jackjosh211@gmail.com";
const client = new SESClient();


//Created function
module.exports.sendEmail = async (event, context, callback) => {
    console.log("EVENT::: Sending mail");


    //Body grabbed from JSON event and parsed
    const { emailAddress, empName, reportText, projects, submittedAt } = JSON.parse(event.body);


    //Validation check
    if(typeof empName !== "string" || typeof reportText !== "string") {
        console.error("Validation Failed");
        return;
    }
    console.log("Email data grabbed and parsed", event.body);


    //Declare parameters to be sent to email
    const emailText = {
        Source: EMAIL,

        Destination: {
            ToAddresses: [
                EMAIL
            ]
        },

        Message: {
            Subject: {
                Data: `${empName}'s (${submittedAt}) Report`
            },

            Body: {
                Text: {
                    Data: `Submitter: ${empName} (${submittedAt})\n\n\nProjects: ${projects}\nReport Text: ${reportText}\n\n\n\n\nEmployee Email: ${emailAddress}\nEmployee ID: ${event.pathParameters.empid}\nReport ID: ${event.pathParameters.reportid}`
                }
            }
        }
    };
```

```
    try {
        //Send parameters to email
        await client.send(new SendEmailCommand(emailText));

        //Response sent to API Gateway
        const response = {
            isBase64Encoded: false,
            statusCode: 200,
            headers: {
                'Content-Type': 'application/json',
                'Access-Control-Allow-Origin': '*'
            },
            body: JSON.stringify(reportText)
        }

        console.log("SUCCESS::: Email sent to " + EMAIL)
        callback(null, response);
    } catch(err) {
        callback(null, err);
    }
};
```



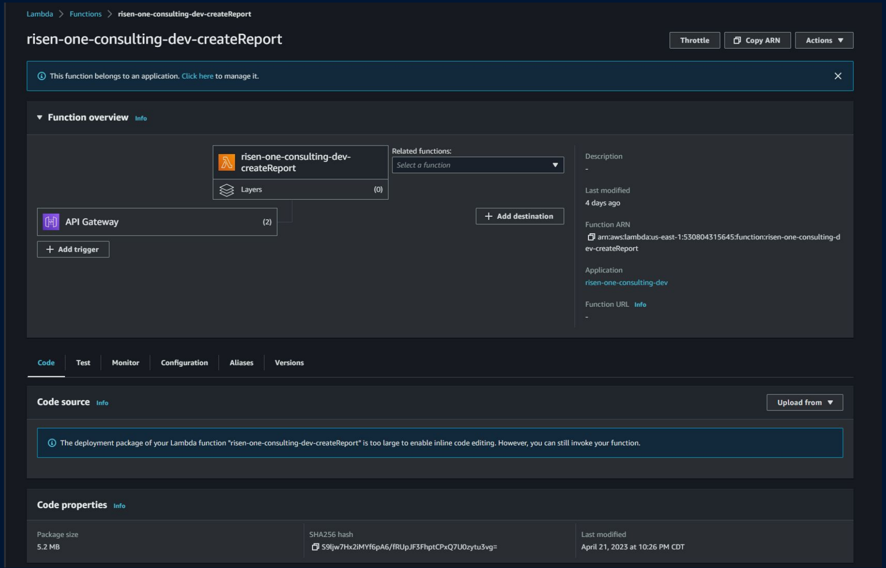
to me

Submitter: Jackson Boster (04/25/2023)

Projects: FLP
Report Text: FLP is going well

Employee Email: jcb73540@ucmo.edu
Employee ID: 929fb744-264b-4e44-afcf-a4b051762fe8
Report ID: b54d800a-d420-4514-a662-35d9c56345d4

Back End Explanation: Connection



Lambda > Functions > risen-one-consulting-dev-createReport

risen-one-consulting-dev-createReport

This function belongs to an application. Click here to manage it.

Function overview Info

Description
-

Last modified
4 days ago

Function ARN
arn:aws:lambda:us-east-1:530804315645:function:risen-one-consulting-dev-createReport

Application
risen-one-consulting-dev

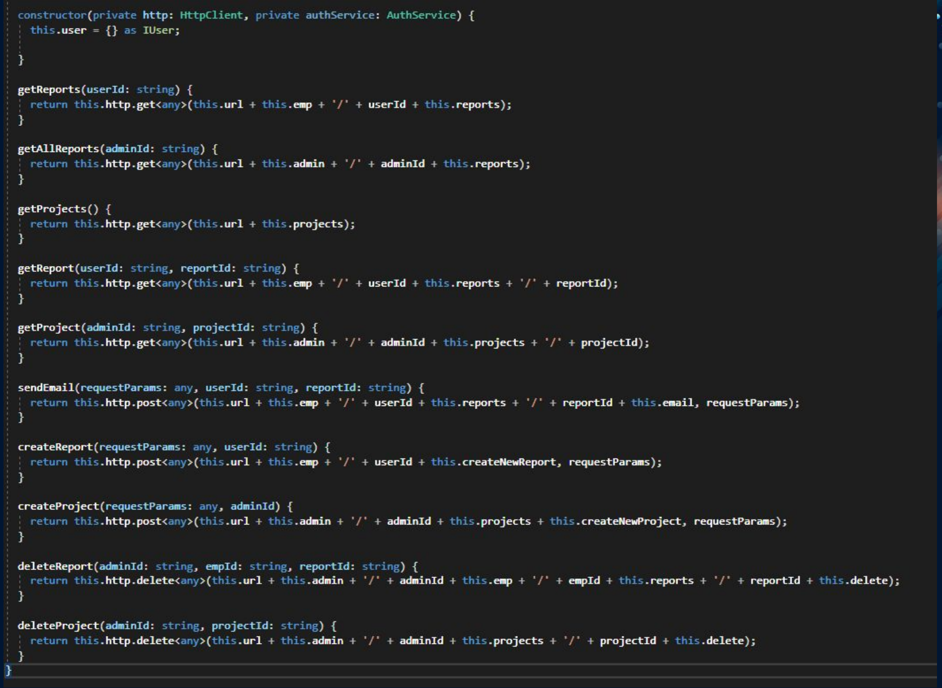
Function URL Info
-

Code source Info

The deployment package of your Lambda function "risen-one-consulting-dev-createReport" is too large to enable inline code editing. However, you can still invoke your function.

Code properties Info

Package size	SHA256 hash	Last modified
5.2 MB	S9ljw7Hx2iMYf6pA6/fRUpJF3FhptCPxQ7U0zytu3vg=	April 21, 2023 at 10:26 PM CDT

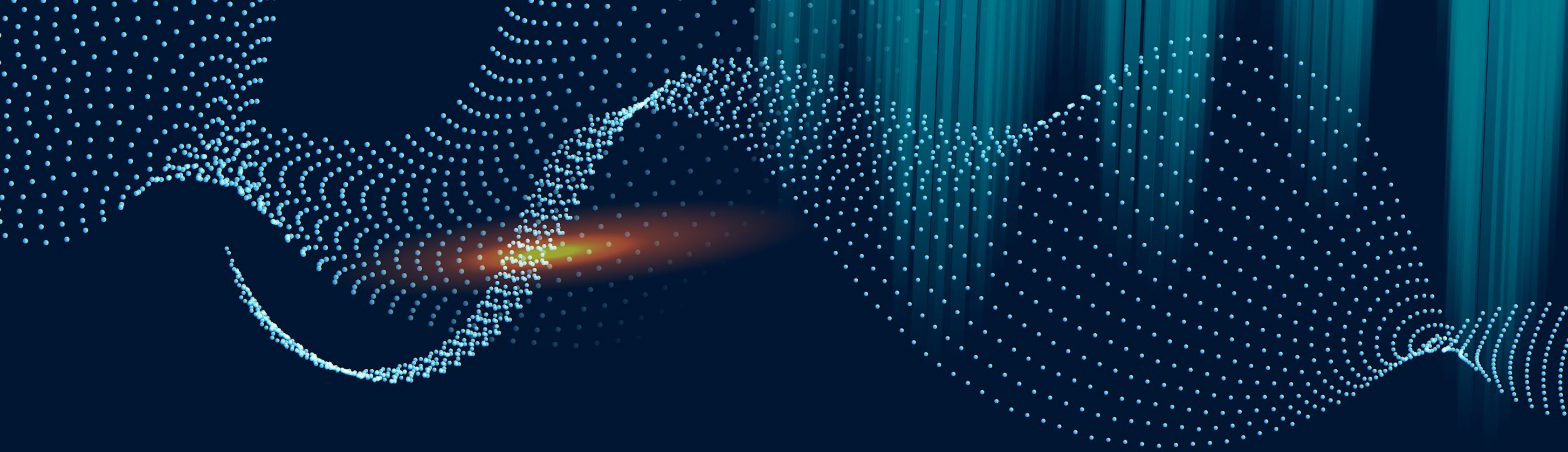


```
constructor(private http: HttpClient, private authService: AuthService) {
  this.user = {} as IUser;

}

getReports(userId: string) {
  return this.http.get<any>(this.url + this.emp + '/' + userId + this.reports);
}

getAllReports(adminId: string) {
  return this.http.get<any>(this.url + this.admin + '/' + adminId + this.reports);
}

getProjects() {
  return this.http.get<any>(this.url + this.projects);
}

getReport(userId: string, reportId: string) {
  return this.http.get<any>(this.url + this.emp + '/' + userId + this.reports + '/' + reportId);
}

getProject(adminId: string, projectId: string) {
  return this.http.get<any>(this.url + this.admin + '/' + adminId + this.projects + '/' + projectId);
}

sendEmail(requestParams: any, userId: string, reportId: string) {
  return this.http.post<any>(this.url + this.emp + '/' + userId + this.reports + '/' + reportId + this.email, requestParams);
}

createReport(requestParams: any, userId: string) {
  return this.http.post<any>(this.url + this.emp + '/' + userId + this.createNewReport, requestParams);
}

createProject(requestParams: any, adminId) {
  return this.http.post<any>(this.url + this.admin + '/' + adminId + this.projects + this.createNewProject, requestParams);
}

deleteReport(adminId: string, empId: string, reportId: string) {
  return this.http.delete<any>(this.url + this.admin + '/' + adminId + this.emp + '/' + empId + this.reports + '/' + reportId + this.delete);
}

deleteProject(adminId: string, projectId: string) {
  return this.http.delete<any>(this.url + this.admin + '/' + adminId + this.projects + '/' + projectId + this.delete);
}
}
```

Struggles through Sprints

- Jira
    - Group had acute communication
    - Struggled with using external software to track process

- User Pools
    - Underestimated the difficulty of implementing Cognito user pools which
    - Led to staying on this issue for longer than desired



Questions?