Week 5 submission

Project Title: ” HR Management System”

Domain: ” Cloud Computing”

Prepared by

1. **Nensi Ravaliya**
2. **Priti Kumari**
3. **Bhagyashri Saundarkar**

Summary of week-5 progress

Objective:

The Human Resource Management System (HRMS) aims to manage employee's information. The objective of a Human Resources (HR) Management System is to streamline and optimize various HR-related processes within an organization. It is a software solution that helps HR departments manage their tasks more efficiently, automate routine processes, and improve overall HR operations.

Overall, the objective of an HR Management System is to enhance HR efficiency, improve employee experiences, support compliance with regulations, and contribute to the organization's overall success by managing human capital effectively.

Problem Statement Understood: YES

Completed Code Implementation (in percentage): 90%

[HR Management System]

Brief about Design/ use case study:

The Human Resource Management System (HRMS) aims to manage employee's information. HRMS is the most important because it provides systematic and accurate information about the employees of the organization.

The primary key features design of an HR Management System include:

1. Automation: HR systems automate repetitive tasks such as payroll processing, leave management, attendance tracking, and benefits administration. This reduces manual errors and frees up HR staff to focus on more strategic tasks.

2. Recruitment and Onboarding: The system assists in the recruitment process by posting job openings, receiving applications, managing candidate information, and facilitating onboarding processes for new hires.

3. Performance Management: HR systems often include tools for setting and tracking performance goals, conducting regular performance evaluations, and providing feedback to employees.

4. Training and Development: The system helps in planning, managing, and tracking employee training and development programs, ensuring that employees acquire the necessary skills for their roles.

5. Employee Self-Service: Many HR systems offer employee self-service portals where employees can access their personal information, request leaves, update details, and access relevant HR policies and documents.

6. Compliance and Reporting: HR systems assist in ensuring compliance with labor laws and regulations by generating reports related to workforce diversity, equal opportunity employment, payroll taxes, and more.

7. Benefits Administration: The system facilitates benefits enrolment, tracks employee benefit choices, and manages processes related to health insurance, retirement plans, and other benefits.

8. Succession Planning: HR systems can help organizations identify potential future leaders and create succession plans to ensure a smooth transition of key roles.

9. Analytics and Decision-Making: HR systems often include reporting and analytics features that allow HR professionals to generate insights from the data, enabling informed decision-making and strategic planning.

10. Employee Engagement: Some HR systems provide tools for measuring employee engagement, satisfaction, and feedback, which can help organizations improve their work environment and employee morale.

11. Cost Savings: By automating various HR tasks, organizations can save time, reduce administrative costs, and minimize errors that can arise from manual processes.

Technology:

Frontend: HTML, CSS, JavaScript, React.js, java.

Database: DynamoDB

Services: AWS lambda, AWS CloudFront, EC2, S3 bucket etc.

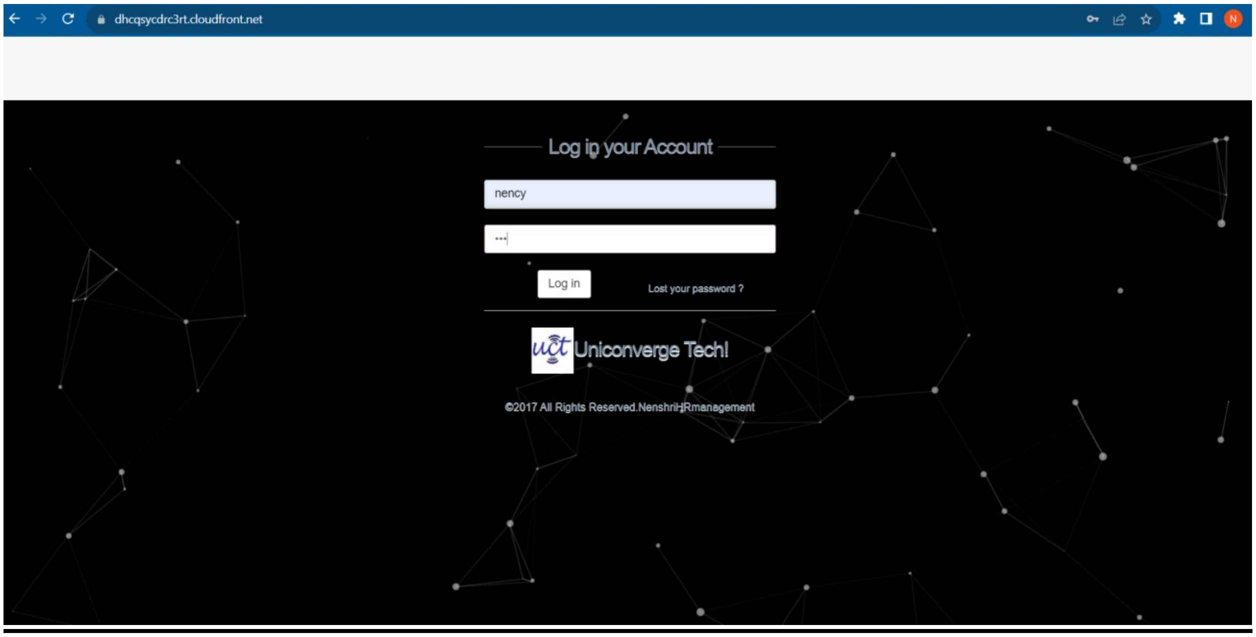
In this week we create table in AWS DynamoDB database. Also, there are two lamda functions for getting the data from database and Insert data into database.

The objective of this project was to create a dynamic website that could store and retrieve data from a DynamoDB database using AWS Lambda functions. The deployment was performed in several phases, including:

1. Developing the frontend of the website.
2. Hosting the website on AWS cloud.
3. Implementing content delivery through CloudFront.
4. Setting up a DynamoDB database.
5. Creating Lambda functions for data retrieval and insertion.
6. Creating an API to access data from the website.

Website link: https://dhcqsycdrc3rt.cloudfront.net

Implementation on AWS:



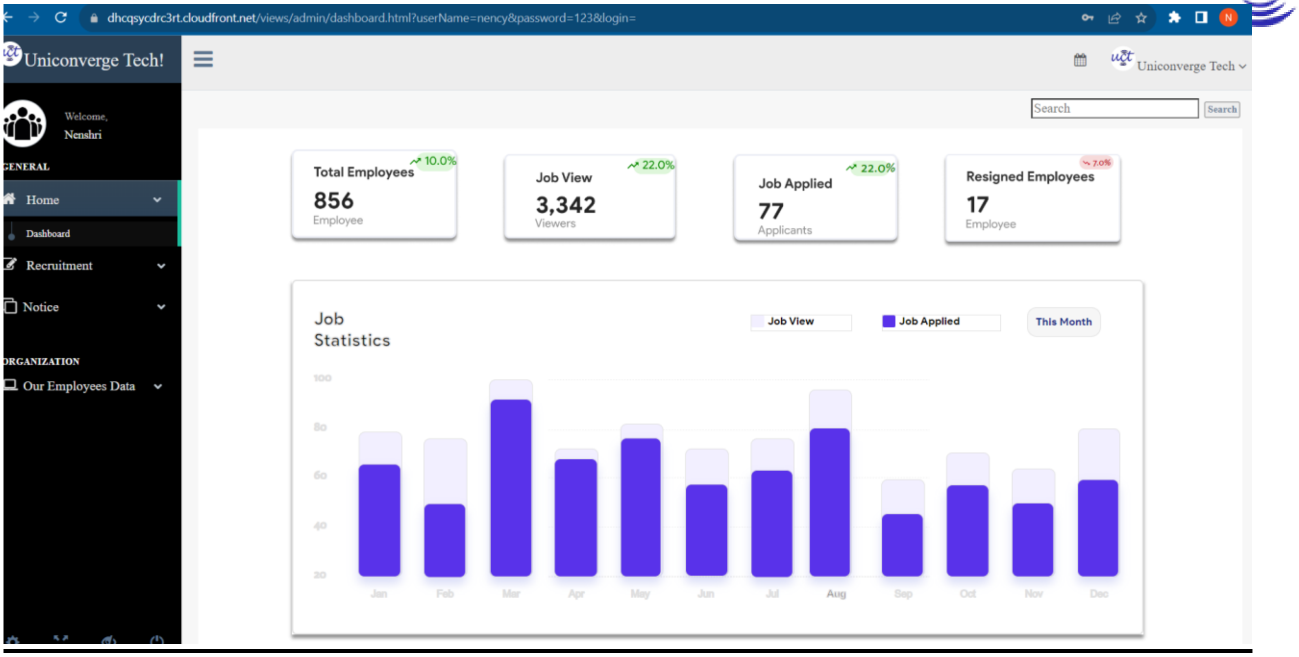
dhcqsycdrc3rt.cloudfront.net

Log ip your Account

nency

•••

©2017 All Rights Reserved.NenshriHRmanagement



dhcqsycdrc3rt.cloudfront.net/views/admin/dashboard.html?userName=nency&password=123&login=

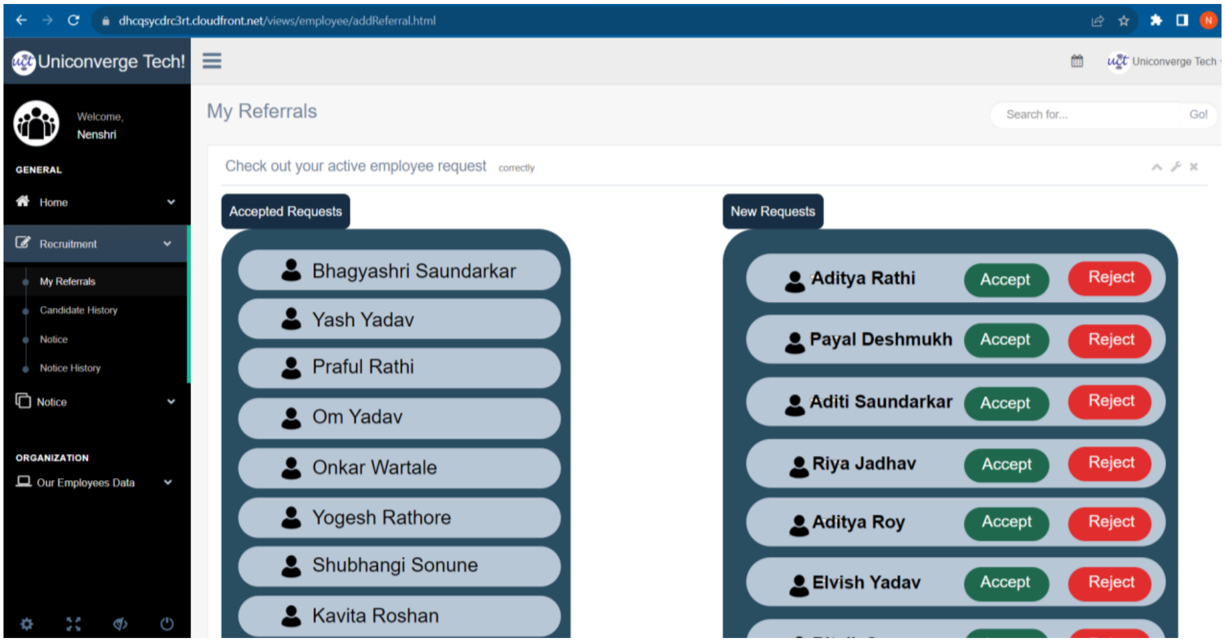
Total Employees 10.0%
856
Employee

Job View 22.0%
3,342
Viewers

Job Applied 22.0%
77
Applicants

Resigned Employees 7.0%
17
Employee

Job Statistics



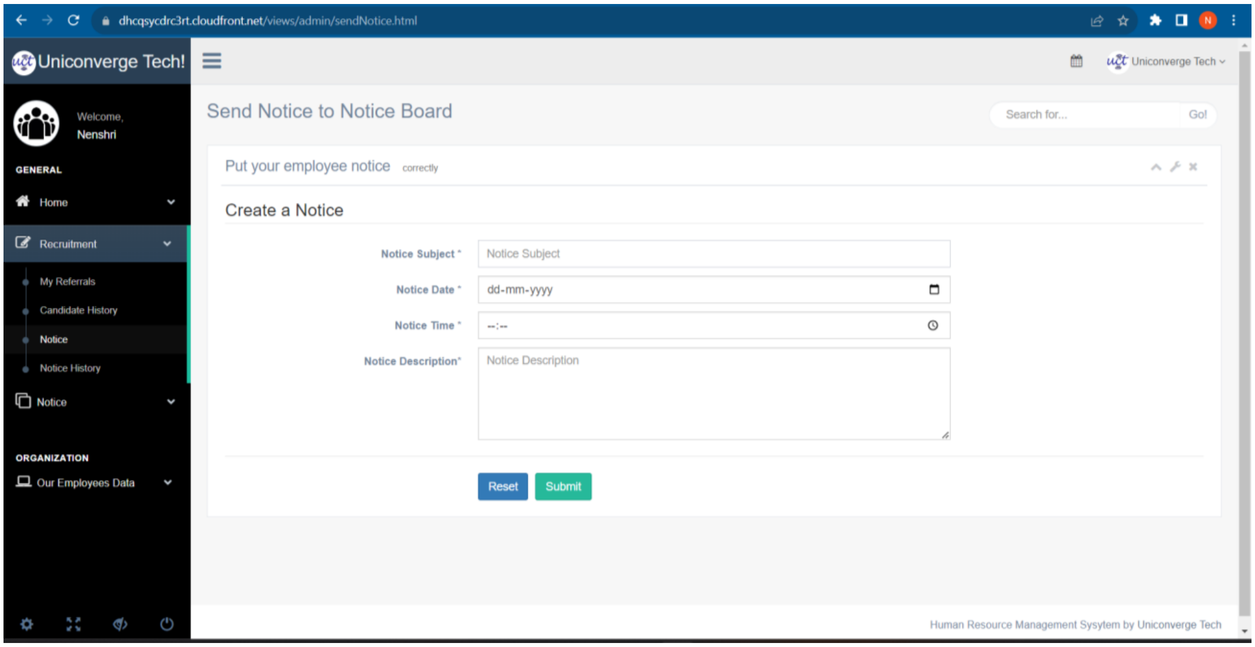
dhcqsycdrc3rt.cloudfront.net/views/employee/addReferral.html

My Referrals

Check out your active employee request correctly

Accepted Requests
Bhagyashri Saundarkar
Yash Yadav
Praful Rathi
Om Yadav
Onkar Wartale
Yogesh Rathore
Shubhangi Sonune
Kavita Roshan

New Requests		
Aditya Rathi	Accept	Reject
Payal Deshmukh	Accept	Reject
Aditi Saundarkar	Accept	Reject
Riya Jadhav	Accept	Reject
Aditya Roy	Accept	Reject
Elvish Yadav	Accept	Reject

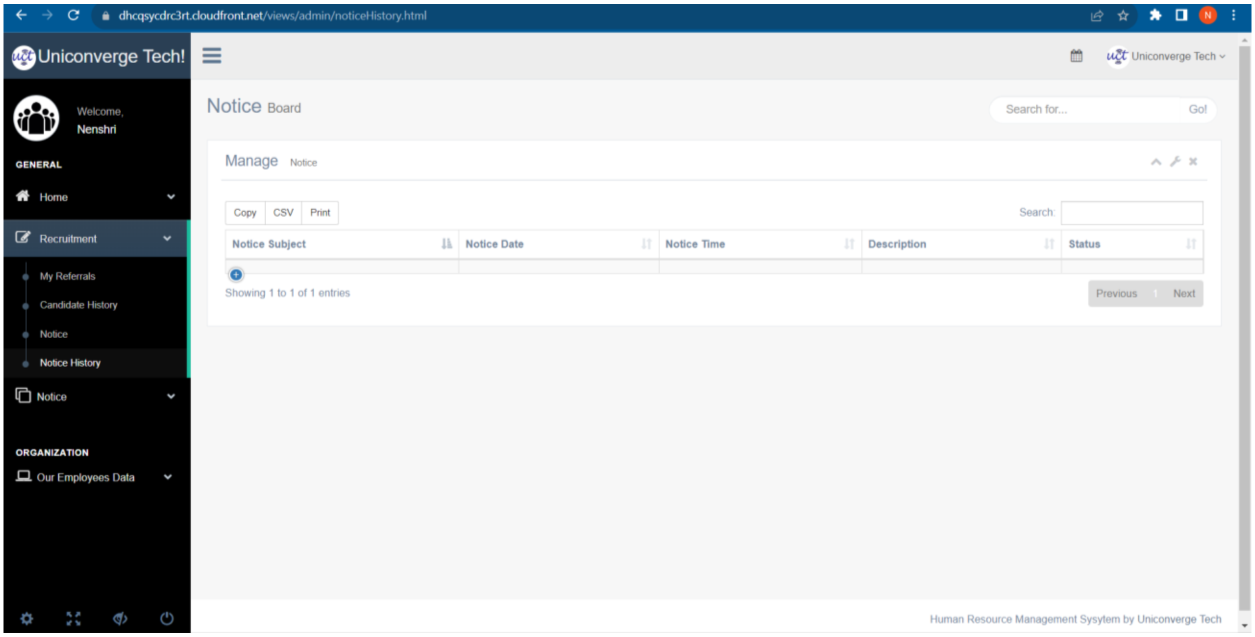


Send Notice to Notice Board

Put your employee notice correctly

Create a Notice

Notice Subject *	Notice Subject
Notice Date *	dd-mm-yyyy
Notice Time *	--:--
Notice Description*	Notice Description

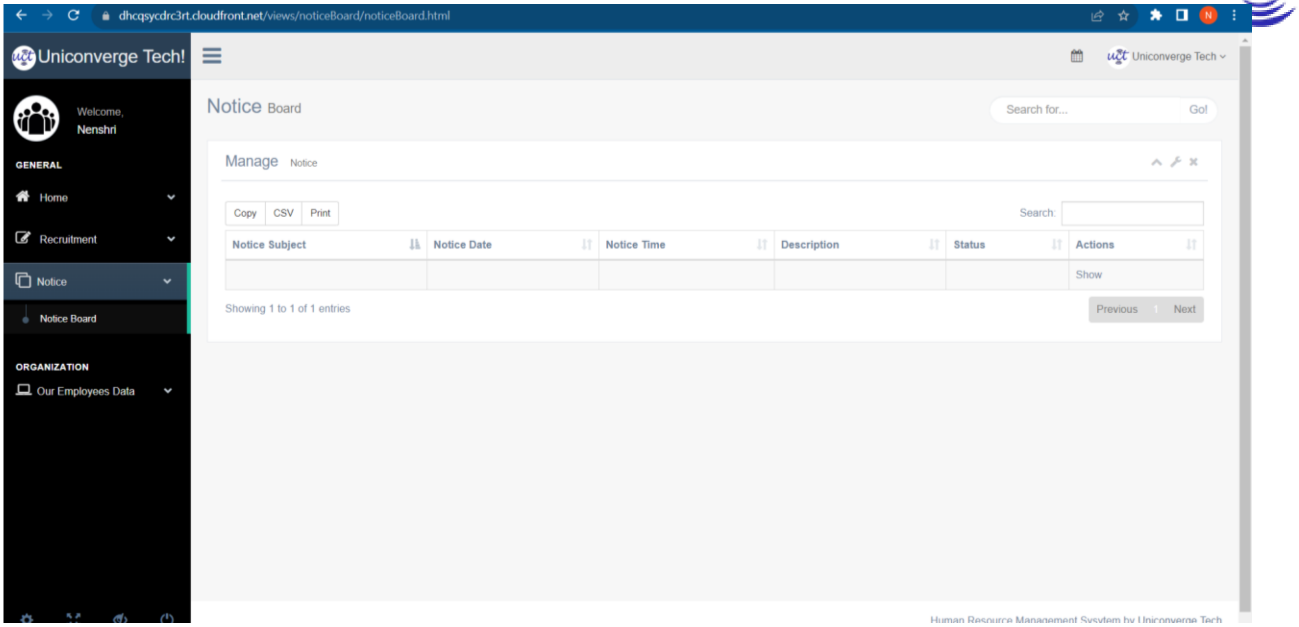


Notice Board

Manage Notice

Notice Subject	Notice Date	Notice Time	Description	Status

Showing 1 to 1 of 1 entries



dhcqsycdrc3rt.cloudfront.net/views/noticeBoard/noticeBoard.html

Uniconverge Tech!

Welcome,
Nenshri

GENERAL

Home

Recruitment

Notice

Notice Board

ORGANIZATION

Our Employees Data

Uniconverge Tech

Notice Board

Search for... Go!

Manage Notice

Copy CSV Print

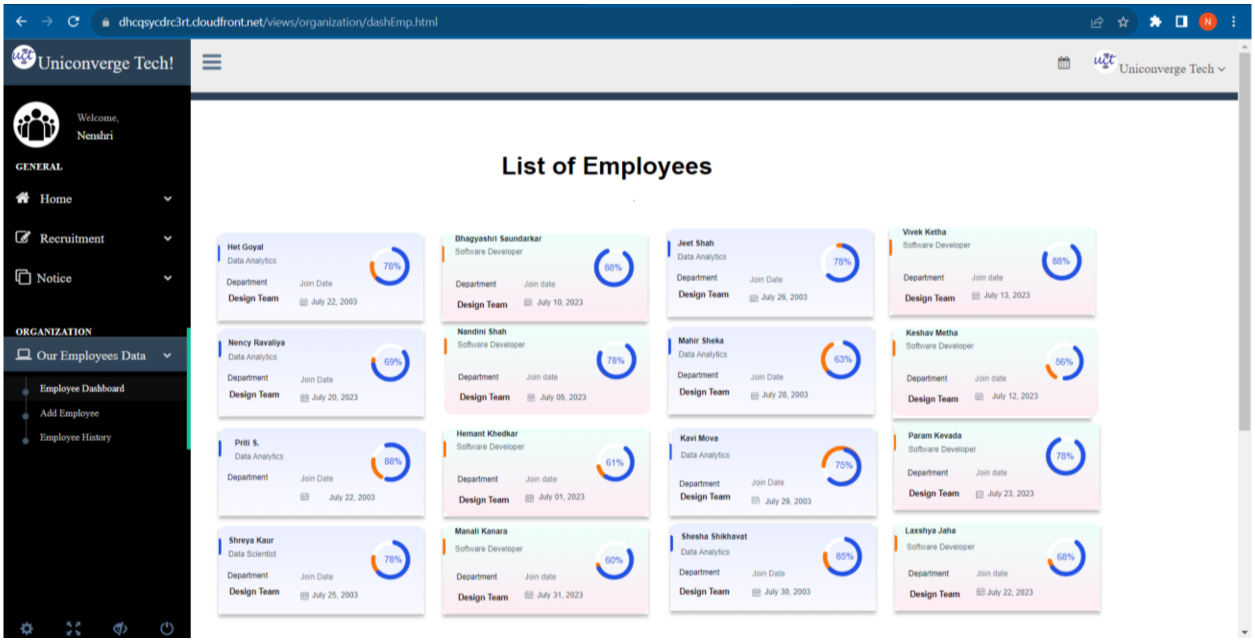
Search:

Notice Subject	Notice Date	Notice Time	Description	Status	Actions
					Show

Showing 1 to 1 of 1 entries

Previous 1 Next

Human Resource Management System by Uniconverge Tech



dhcqsycdrc3rt.cloudfront.net/views/organization/dashEmp.html

Uniconverge Tech!

Welcome,
Nenshri

GENERAL

Home

Recruitment

Notice

ORGANIZATION

Our Employees Data

Employee Dashboard

Add Employee

Employee History

Uniconverge Tech

List of Employees

Het Goyal
Data Analytics
78%
Department Join Date
Design Team July 22, 2003

Bhagyashri Saundarkar
Software Developer
88%
Department Join date
Design Team July 10, 2023

Jeet Shah
Data Analytics
78%
Department Join Date
Design Team July 26, 2003

Vivek Ketha
Software Developer
88%
Department Join date
Design Team July 13, 2023

Nency Ravaliya
Data Analytics
69%
Department Join Date
Design Team July 20, 2023

Nandini Shah
Software Developer
78%
Department Join date
Design Team July 05, 2023

Mahir Sheka
Data Analytics
63%
Department Join Date
Design Team July 28, 2003

Keshav Metha
Software Developer
56%
Department Join date
Design Team July 12, 2023

Priti S.
Data Analytics
88%
Department Join Date
July 22, 2003

Hemant Khedkar
Software Developer
61%
Department Join date
Design Team July 01, 2023

Kavi Mova
Data Analytics
75%
Department Join Date
Design Team July 29, 2003

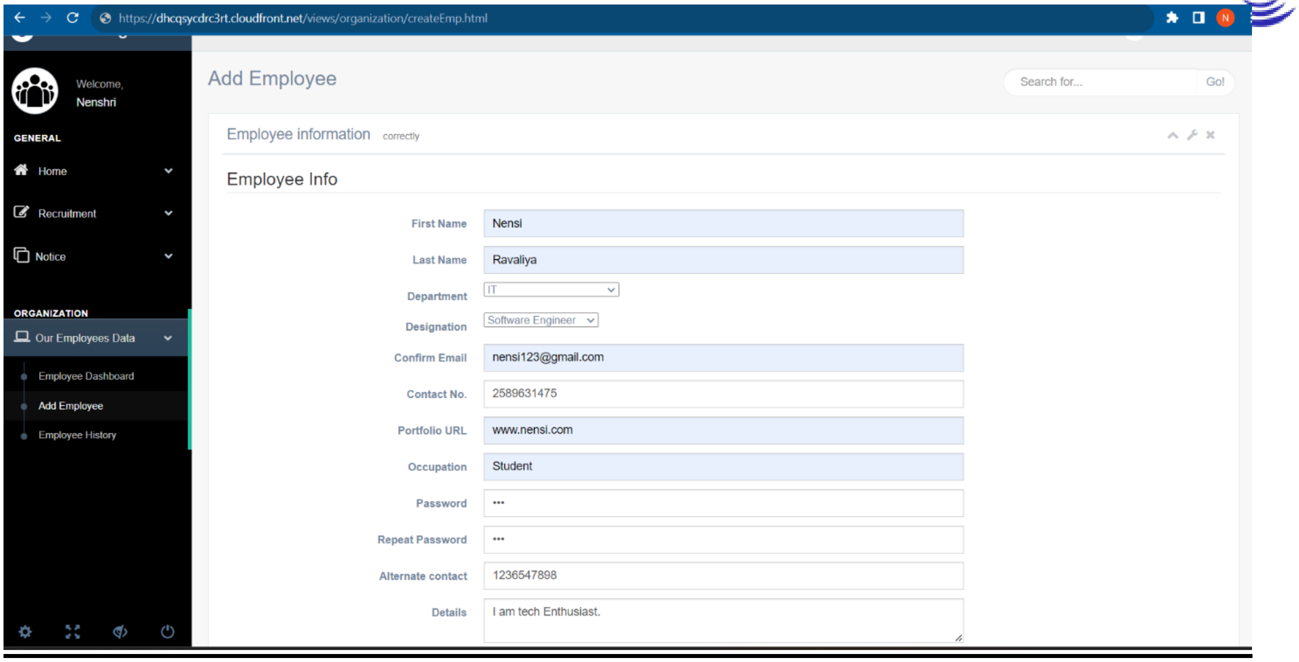
Param Kevada
Software Developer
78%
Department Join date
Design Team July 23, 2023

Shreya Kaur
Data Scientist
78%
Department Join Date
Design Team July 25, 2003

Manali Kanara
Software Developer
60%
Department Join date
Design Team July 31, 2023

Shesha Shikhavat
Data Analytics
85%
Department Join Date
Design Team July 30, 2003

Laxshya Jaha
Software Developer
68%
Department Join date
Design Team July 22, 2023

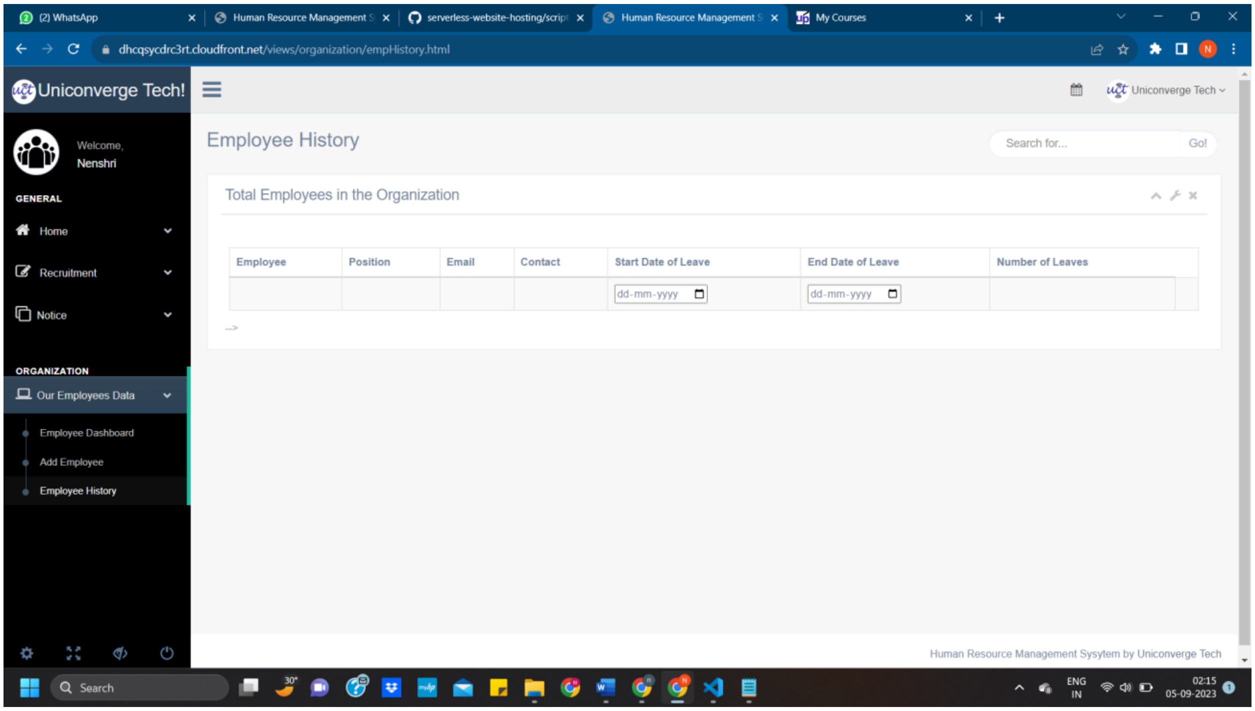


Add Employee

Employee information correctly

Employee Info

First Name	Nensi
Last Name	Ravaliya
Department	IT
Designation	Software Engineer
Confirm Email	nensi123@gmail.com
Contact No.	2589631475
Portfolio URL	www.nensi.com
Occupation	Student
Password	•••
Repeat Password	•••
Alternate contact	1236547898
Details	I am tech Enthusiast.



Employee History

Total Employees in the Organization

Employee	Position	Email	Contact	Start Date of Leave	End Date of Leave	Number of Leaves
				dd-mm-yyyy	dd-mm-yyyy	

-->

1. Frontend Development

The project commenced with the development of the website's frontend. This phase involved creating the user interface and designing the website's layout and functionalities.

2. AWS Cloud Hosting

The website was hosted on AWS using the following steps:

An AWS EC2 instance was launched to host the website's backend and associated services.

The website's frontend files were deployed to an Amazon S3 bucket.

AWS Identity and Access Management (IAM) policies were configured to secure access to AWS resources.

3. CloudFront Distribution

To optimize content delivery and enhance website performance, a CloudFront distribution was created:

A CloudFront distribution was set up to cache and deliver content from S3.

DNS records were updated to point to the CloudFront distribution for website access.

4. DynamoDB Database

A DynamoDB database was provisioned to store and manage data for the website:

A DynamoDB table was created with the necessary schema to store data.

Appropriate read and write capacity units were allocated based on expected usage.

5. Lambda Functions

Two Lambda functions were created to interact with the DynamoDB database:

Get Function: This Lambda function was designed to retrieve data from the DynamoDB table.

Insert Function: This Lambda function was responsible for inserting data into the DynamoDB table.

6. API Gateway

API Gateway was utilized to expose the Lambda functions as RESTful APIs:

API Gateway endpoints were created to trigger the Lambda functions.

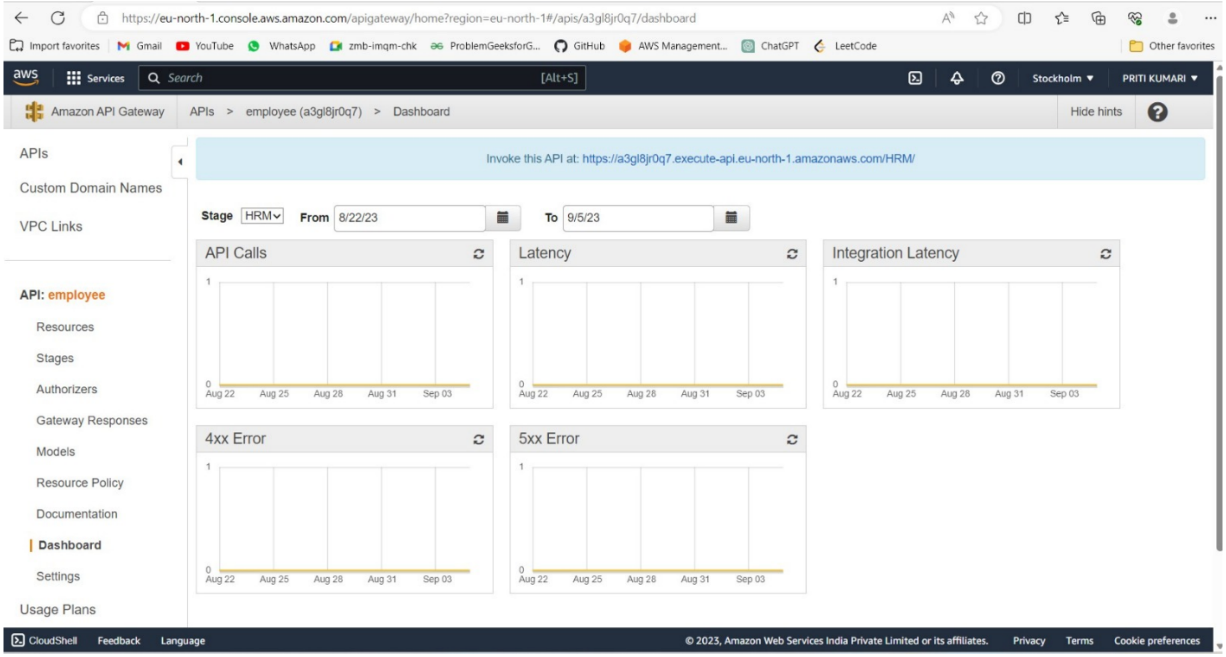
Appropriate authentication and authorization mechanisms were configured for API security.

7. Website Access

With all components in place, the website was made accessible through a web browser:

Users could access the website by entering the URL in their browsers.

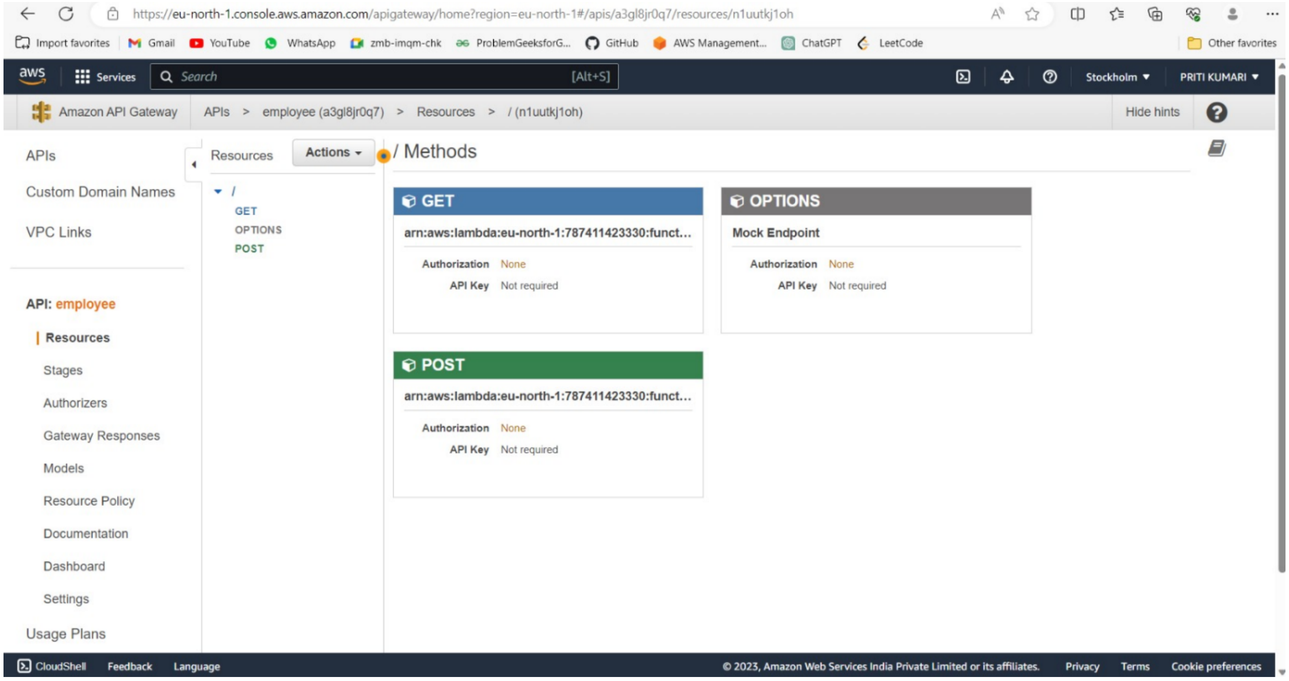
The website interacted with the API endpoints to retrieve and display data from DynamoDB.



https://eu-north-1.console.aws.amazon.com/apigateway/home?region=eu-north-1#/apis/a3gl8jr0q7/dashboard

Invoke this API at: https://a3gl8jr0q7.execute-api.eu-north-1.amazonaws.com/HRM/

Stage HRM From 8/22/23 To 9/5/23



https://eu-north-1.console.aws.amazon.com/apigateway/home?region=eu-north-1#/apis/a3gl8jr0q7/resources/n1uutkj1oh

Amazon API Gateway > APIs > employee (a3gl8jr0q7) > Resources > / (n1uutkj1oh)

API: employee

/ Methods

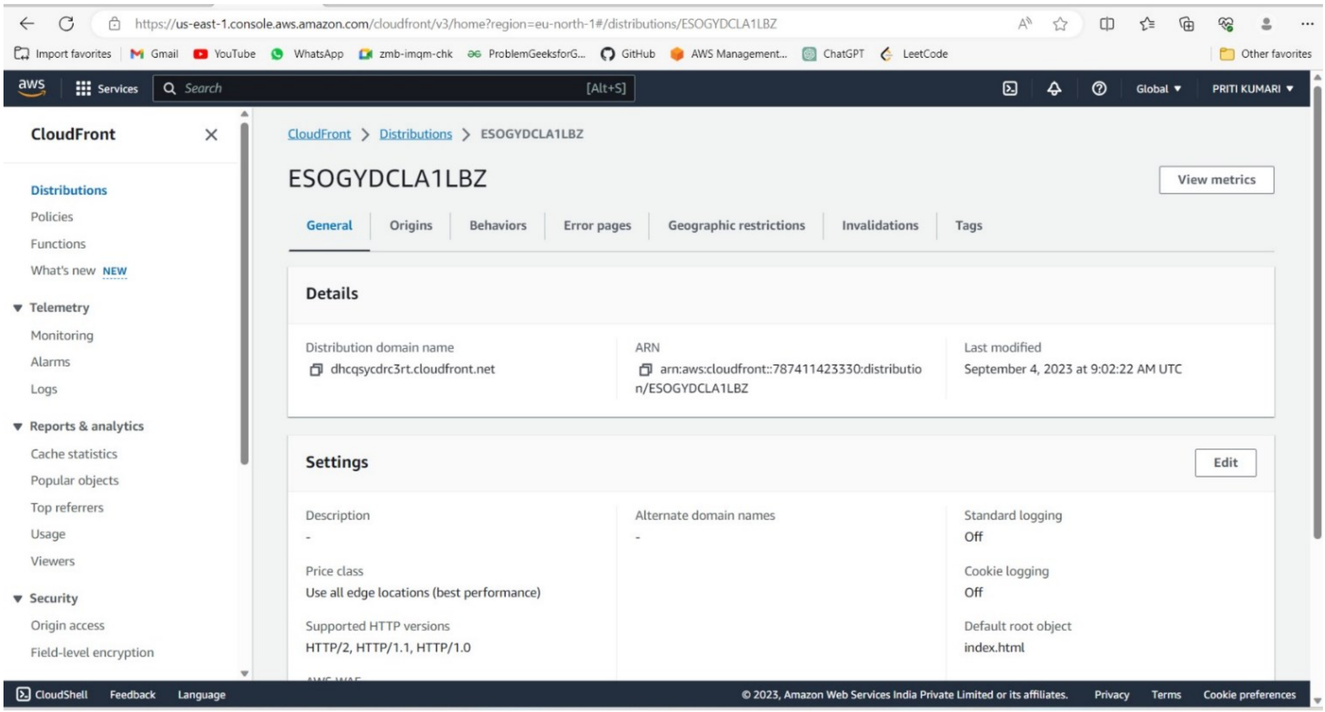
Resources:
- /
  - GET
  - OPTIONS
  - POST

GET
arn:aws:lambda:eu-north-1:787411423330:funct...
Authorization None
API Key Not required

OPTIONS
Mock Endpoint
Authorization None
API Key Not required

POST
arn:aws:lambda:eu-north-1:787411423330:funct...
Authorization None
API Key Not required

© 2023, Amazon Web Services India Private Limited or its affiliates.



https://us-east-1.console.aws.amazon.com/cloudfront/v3/home?region=eu-north-1#/distributions/ESOGYDCLA1LBZ

CloudFront > Distributions > ESOGYDCLA1LBZ

ESOGYDCLA1LBZ

Details

Distribution domain name: dhcqsycdrc3rt.cloudfront.net

ARN: arn:aws:cloudfront::787411423330:distribution/ESOGYDCLA1LBZ

Last modified: September 4, 2023 at 9:02:22 AM UTC

Settings

Description: -

Alternate domain names: -

Standard logging: Off

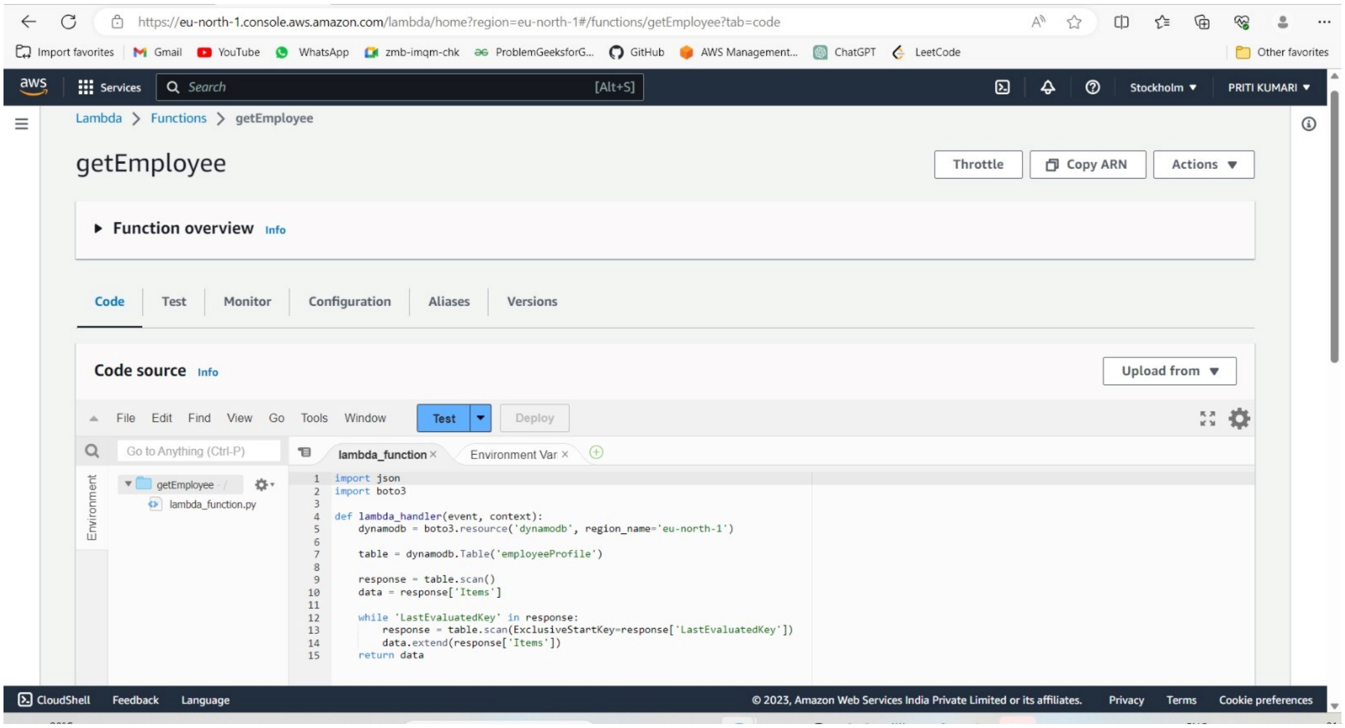
Price class: Use all edge locations (best performance)

Cookie logging: Off

Supported HTTP versions: HTTP/2, HTTP/1.1, HTTP/1.0

Default root object: index.html

© 2023, Amazon Web Services India Private Limited or its affiliates.



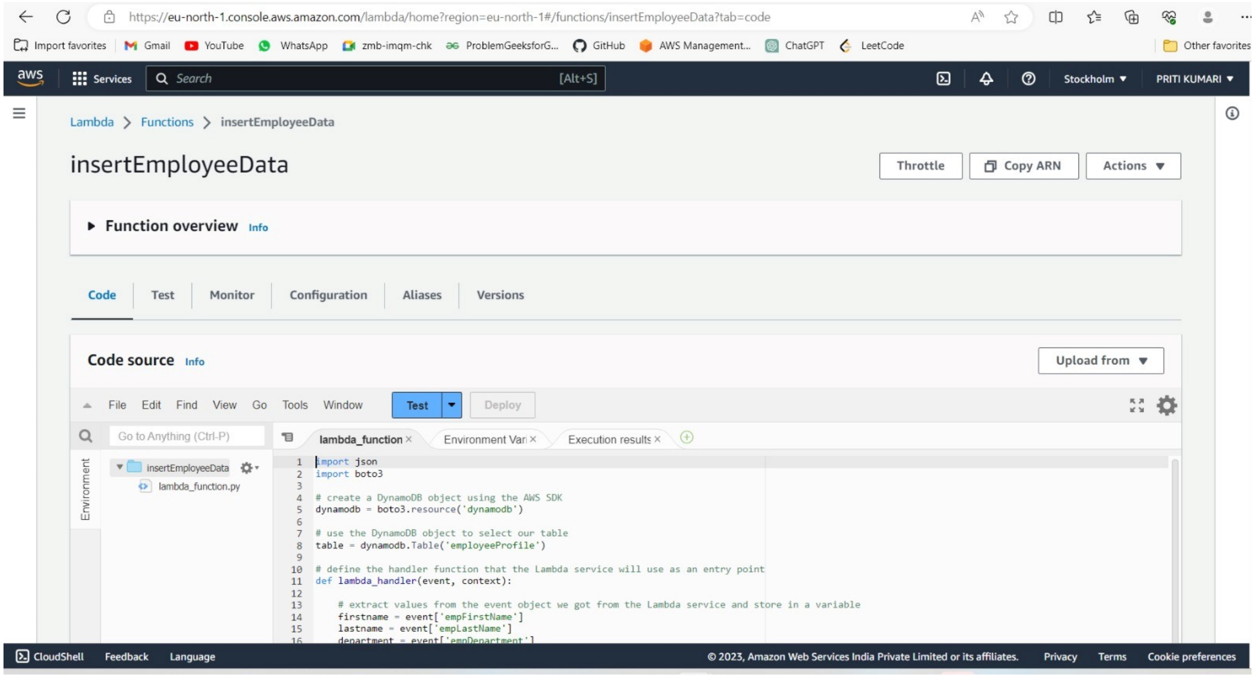
```python
import json
import boto3

def lambda_handler(event, context):
    dynamodb = boto3.resource('dynamodb', region_name='eu-north-1')

    table = dynamodb.Table('employeeProfile')

    response = table.scan()
    data = response['Items']

    while 'LastEvaluatedKey' in response:
        response = table.scan(ExclusiveStartKey=response['LastEvaluatedKey'])
        data.extend(response['Items'])
    return data
```



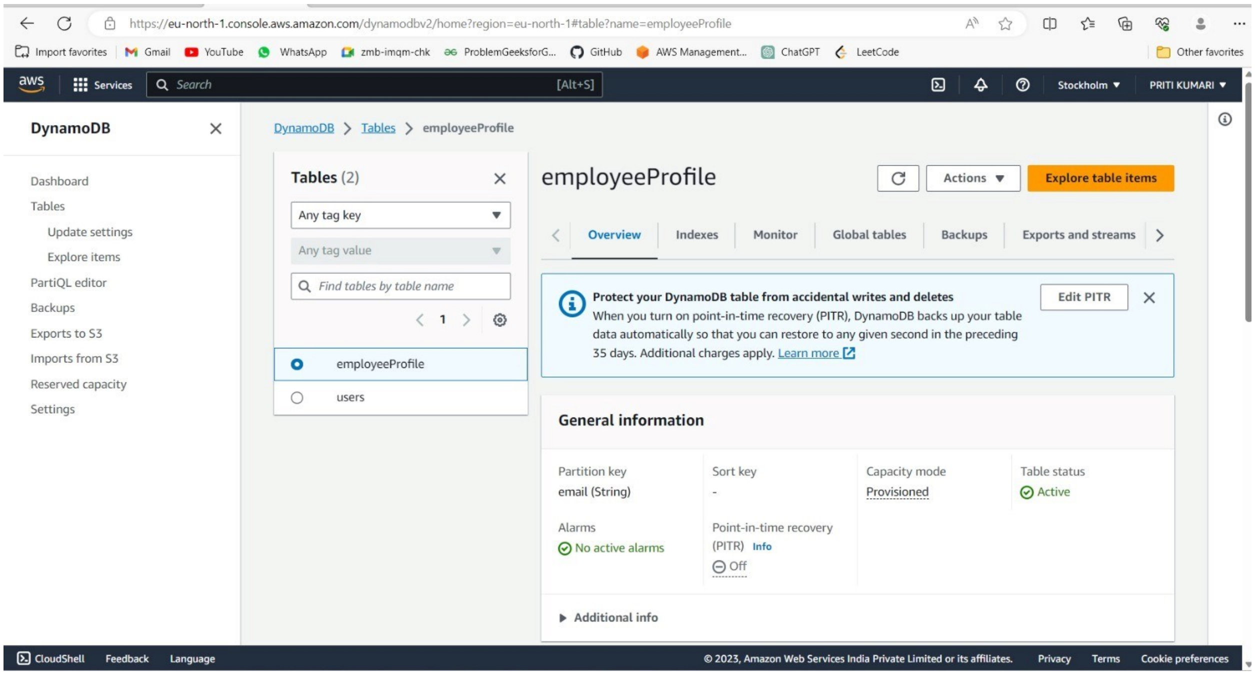
```python
import json
import boto3

# create a DynamoDB object using the AWS SDK
dynamodb = boto3.resource('dynamodb')

# use the DynamoDB object to select our table
table = dynamodb.Table('employeeProfile')

# define the handler function that the Lambda service will use as an entry point
def lambda_handler(event, context):

    # extract values from the event object we got from the Lambda service and store in a variable
    firstname = event['empFirstName']
    lastname = event['empLastName']
    department = event['empDepartment']
```



● employeeProfile

○ users

employeeProfile

Protect your DynamoDB table from accidental writes and deletes
When you turn on point-in-time recovery (PITR), DynamoDB backs up your table data automatically so that you can restore to any given second in the preceding 35 days. Additional charges apply. Learn more

General information

Partition key: email (String)

Sort key: -

Capacity mode: Provisioned

Table status: Active

Alarms: No active alarms

Point-in-time recovery (PITR) Info: Off

Your Learning Highlights:

1. CloudFront
2. Lamda function
3. IAM policy
4. DynamoDB Database
5. Creating API
6. S3 bucket
7. Bucket Policy
8. Public speaking
9. Soft skills: Communication skill

Problems Faced:

1. Creating and deploying lamda function
2. Understanding about ClouFrontCreating and really understating of S3 bucket in AWS cloud is different then we Imagin.
3. Creating Policy for bucket is add adjust it according to our website is require more effort and deep understanding of AWS cloud.
4. CloudFront for hosting website with AWS S3 bucket.
5. Technical Challenges:

   i). Integration Issues: Ensuring seamless integration between various AWS services, such as AWS S3, CloudFront, DynamoDB, Lambda, and API Gateway, can be complex and may require thorough testing and debugging.

   ii). Performance Optimization: Fine-tuning the website's performance, especially in terms of response times and latency, may be necessary to ensure a smooth user experience.

6. **Security Concerns:**

   Data Security: Protecting sensitive data in transit and at rest, such as user information in the DynamoDB database, is paramount. Security misconfigurations could lead to data breaches.

   Access Control: Managing AWS IAM policies and roles correctly to restrict access to authorized personnel is critical. Misconfigurations here could result in unauthorized access or data leaks.

7. **Scalability and Cost Management:**

i). Resource Scaling: Monitoring and scaling resources to meet changing traffic patterns and demands is essential. Failure to do so could lead to performance issues during traffic spikes or unnecessary costs during low traffic periods.

ii). Cost Optimization: Without proper monitoring and cost controls, AWS bills can quickly escalate. Ensuring cost-effective resource usage is crucial.

8. **Error Handling and Debugging:**

i). Error Handling: Handling errors and exceptions gracefully, especially in Lambda functions and API Gateway, is crucial for a reliable user experience.

ii). Debugging: Identifying and resolving issues within the AWS environment, such as Lambda function errors or API Gateway misconfigurations, can be challenging.

9. **Backup and Recovery:**

i). Data Backup: Implementing robust backup and recovery procedures for DynamoDB and other critical resources is necessary to mitigate data loss in case of failures or accidents.

ii). Disaster Recovery: Having a well-defined disaster recovery plan in place can help minimize downtime and data loss in the event of an AWS region outage or other catastrophic events.

10. **Documentation and Knowledge Transfer:**

i). Documentation: Maintaining up-to-date and comprehensive documentation for the AWS architecture and configurations is essential for troubleshooting, scaling, and onboarding new team members.