BULID-A-THON

PROJECT TITLE: EMPLOYEE APPRECIATION BASED ON CUSTOMER SATISFACTION

NAME: BARATHKUMAR G
BANNARI AMMAN INSTITUTE OF TECHNOLOGY

INDEX

1.INTRODUCTION

Employee recognition is the timely, informal or formal acknowledgement of a person's behavior, effort, or business result that supports the organization's goals and values, and exceeds his superior's normal expectations.Recognition has been held to be a constructive response and a judgment made about a person's contribution, reflecting not just work performance but also personal dedication and engagement on a regular or ad hoc basis, and expressed formally or informally, individually or collectively, privately or publicly, and monetarily or non-monetarily

2.LITERATURE SURVEY

2.1Existing problem

How do happier employees make businesses more profitable? The answer is employees who show up every day with passion and purpose are more effective. In fact, teams ranked in the top 20th percentile in engagement see less absenteeism and 59% fewer turnovers. They also make customers more likely to keep coming back.

Leadership that recognizes great employee efforts sets examples for others to follow. Consequently, letting employees nominate peers for recognition can foster camaraderie and make you more aware of low-key stars who don’t shine as brightly, but consistently help their colleagues.

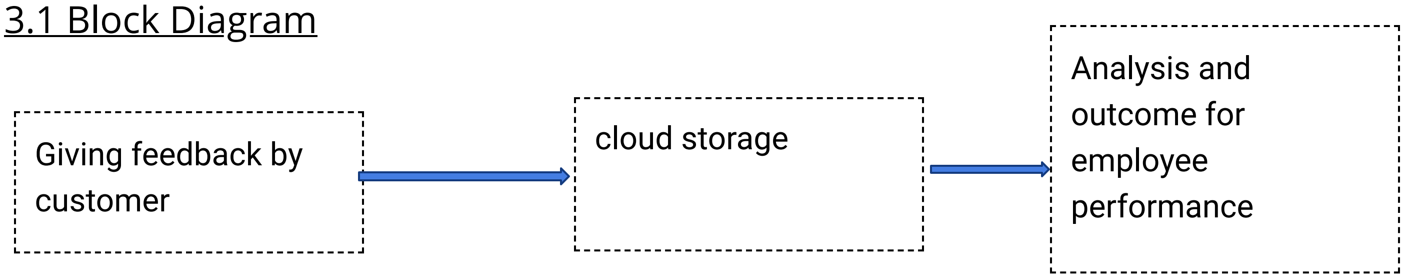
2.2 Proposed solution

When your employees are satisfied with their job, it is going to reflect in the way they handle a customer. They will be able to listen to their concern, become more respectful, and treat them well. Thus, resulting in bonding with a customer well, which in turn will improve your service ratings and an increase in satisfied customers.

We can build the employee score system and integrate with the cloud system to make some analysis and keep track on the performance on the working on the employee.

3.THEORETICAL ANALYSIS

3.1 Block Diagram


Giving feedback by customer
cloud storage
Analysis and outcome for employee performance

3.2 Software Designing

- _ Amazon Comprehend uses natural language processing (NLP) to extract insights about the content of documents. Amazon Comprehend processes any text file in UTF-8 format. It develops insights by recognizing the entities, key phrases, language, sentiments, and other common elements in a document. Use Amazon Comprehend to create new products based on understanding the structure of documents. For example, using Amazon Comprehend you can search social networking feeds for mentions of products or scan an entire document repository for key phrases.

- Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully managed, multiregion, multimaster, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications. DynamoDB can handle more than 10 trillion requests per day and can support peaks of more than 20 million requests per second.

- Many of the world's fastest growing businesses such as Lyft, Airbnb, and Redfin as well as enterprises such as Samsung, Toyota, and Capital One depend on the scale and performance of DynamoDB to support their mission-critical workloads.

4.EXPERIMENTAL INVESTIGATION

Your existing web framework tooling can work seamlessly with the Serverless Framework. Let’s go over how to use the Python web framework Flask to deploy a Serverless REST API.

In this walk-through, we will:

- Deploy a simple API endpoint
- Add a DynamoDB table and two endpoints to create and retrieve a User object
- Set up path-specific routing for more granular metrics and monitoring
- Configure your environment for local development for a faster development experience.

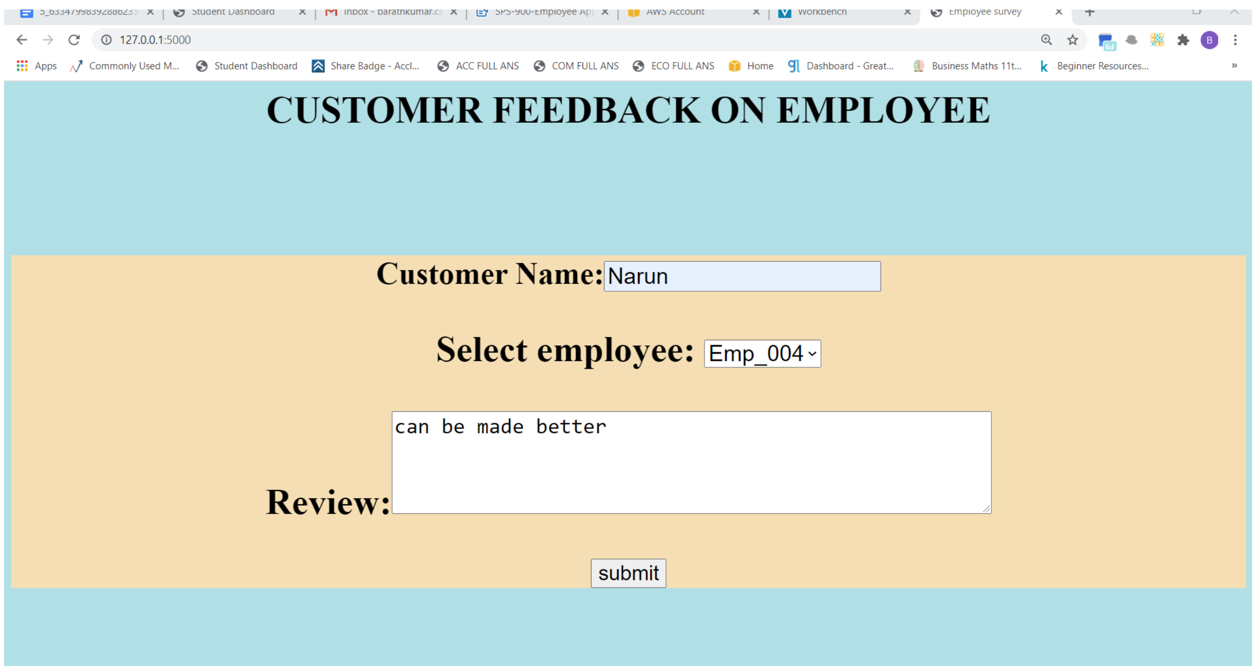
If you already have a Flask application that you want to convert to Serverless, skip to the Converting an existing Flask application

5.FLOWCHART

1. Make the Table in Dynamo db in AWS Cloud service get the accesskey
2. Bulid the simple flask app to get the values for the employee performance
3. Use AWS Comprehend to make the review as positive,neutral or negative and make get stored in the dynamo database.
4. Get the values again from the dynamo database after updating the value for the analysis on the particular employee.
5. Display the result in the graph format.

6.RESULT

- Creating the flask application the output will be



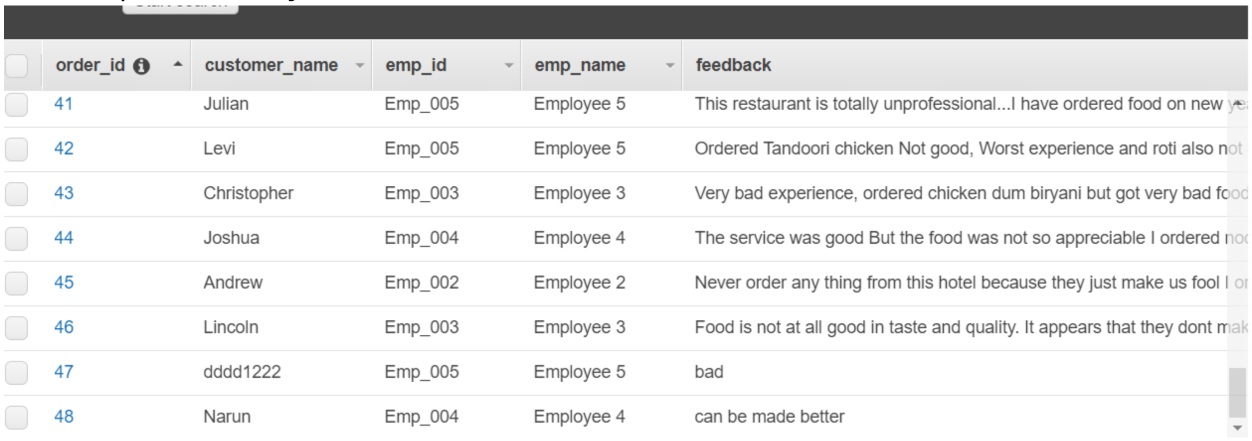
CUSTOMER FEEDBACK ON EMPLOYEE

Customer Name: Narun

Select employee: Emp_004

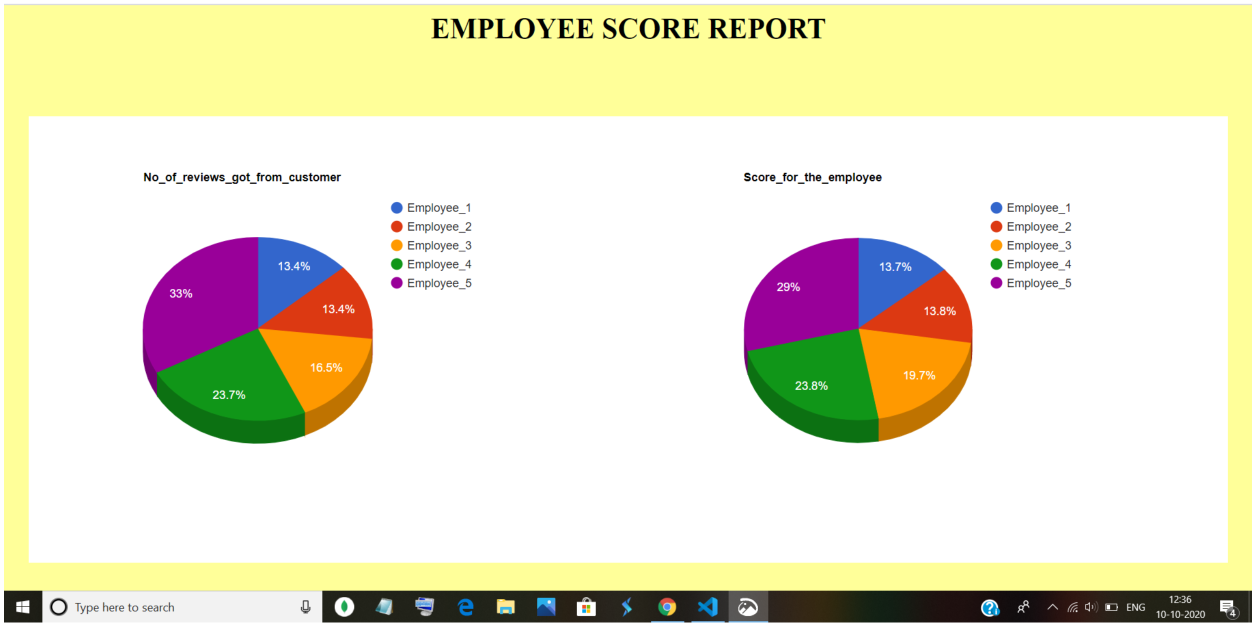
Review: can be made better

- updated in dyanamo db



order_id	customer_name	emp_id	emp_name	feedback
41	Julian	Emp_005	Employee 5	This restaurant is totally unprofessional...I have ordered food on new
42	Levi	Emp_005	Employee 5	Ordered Tandoori chicken Not good, Worst experience and roti also not
43	Christopher	Emp_003	Employee 3	Very bad experience, ordered chicken dum biryani but got very bad food
44	Joshua	Emp_004	Employee 4	The service was good But the food was not so appreciable I ordered
45	Andrew	Emp_002	Employee 2	Never order any thing from this hotel because they just make us fool I
46	Lincoln	Emp_003	Employee 3	Food is not at all good in taste and quality. It appears that they dont
47	dddd1222	Emp_005	Employee 5	bad
48	Narun	Emp_004	Employee 4	can be made better

- Results of employee based o thier score



7.ADVANTAGES

- The statistics can be used to monitor the success of the organization’s recruitment and induction practices.
- Performance appraisal system also helps the management in deciding about the promotions, transfers and rewards of the employee.
- It is easy to identify the under-performers and decide whether you want to keep them hoping for improvement or sometimes have to let them go.
- Both manager and employee, keep performance appraisal records and can retrospectively review the changes in the performance in future.

8.CONCLUSION

Hence this sysyem will improve the quality of service by the employee and makes the interaction between the industries with people about the way of approaching thr customer. The employee scorng system will be work well to improve the standards involved the company.

9.REFERENCE

- https://www.kpi.com/blog
- https://aws.amazon.com/comprehend/

APPENDIX

CODE

- Python flask

```
from flask import Flask,render_template,url_for,request
import os
from credentials import *
import boto3



app = Flask(__name__)
db = boto3.resource('dynamodb',aws_access_key_id = access_key_id,aws_secret_access_key =
secret_access_key,aws_session_token = session_token,region_name = region )
client = boto3.client('dynamodb',aws_access_key_id = access_key_id,aws_secret_access_key =
secret_access_key,aws_session_token = session_token,region_name = region )
table = db.Table("my_table")
final_table = db.Table("my_table2")#employee count table used for updating.(second table)
comprehend_client = boto3.client('comprehend',aws_access_key_id = access_key_id,aws_secret_access_key =
secret_access_key,aws_session_token = session_token,region_name = region )



@app.route('/')
def index():
    return render_template('home.html')

@app.route('/process',methods =['POST'])
def process():
    customer_name = request.form['customer_name']
```

```
employee_id = request.form['emp']
feedback = request.form['review']

if employee_id == 'Emp_001':
    employee_name = "Employee 1"
elif employee_id == "Emp_002":
    employee_name = "Employee 2"
elif employee_id == "Emp_003":
    employee_name = "Employee 3"
elif employee_id == "Emp_004":
    employee_name = "Employee 4"
elif employee_id == "Emp_005":
    employee_name = "Employee 5"

print("The name ofthe customer :",customer_name)
print("The employee id:",employee_id)
print("The feedback is : ",feedback)
print("The name of the employee : ",employee_name)


ItemCount = client.describe_table(TableName='my_table')
no_of_item_in_employee_review = ItemCount['Table']['ItemCount']

items = [no_of_item_in_employee_review,customer_name,feedback,employee_id,employee_name]

table.put_item(
    Item  = {
        'order_id':items[0],
        'customer_name':items[1],
        'feedback':items[2],
        'emp_id':items[3],
        'emp_name':items[4]
    }
)
print("The number  of element in the first data base : ",ItemCount)



get_employee_id = employee_id

count_table_response = final_table.get_item(
    Key={
        'emp_id':get_employee_id
    }
) # it gives the responses of the Employee count table

review_count = count_table_response["Item"]['no_of_reviews']  # second table
score = count_table_response["Item"]['score'] # second table

comprehend_response = comprehend_client.detect_sentiment(Text= feedback,LanguageCode="en")
```

```
result = comprehend_response['Sentiment']

if result == 'POSITIVE':
    final_table.update_item(
        Key={
            'emp_id':get_employee_id
        },
        UpdateExpression='SET no_of_reviews = :val1',

        ExpressionAttributeValues={
            ':val1' : review_count + 1
        }
    )
    final_table.update_item(
        Key={
            'emp_id':get_employee_id
        },
        UpdateExpression='SET score = :val2',

        ExpressionAttributeValues={
            ':val2' : score + 30
        }
    )

elif (result == 'NEGATIVE'):
    final_table.update_item(
        Key={
            'emp_id':get_employee_id
        },
        UpdateExpression='SET no_of_reviews = :val1',

        ExpressionAttributeValues={
            ':val1' : review_count + 1,
        }

    )

    final_table.update_item(
        Key={
            'emp_id':get_employee_id
        },
        UpdateExpression='SET score = :val2',

        ExpressionAttributeValues={
            ':val2' : score + 5
        }
    )

else:
    final_table.update_item(
            Key={
                'emp_id':get_employee_id
            },
            UpdateExpression='SET no_of_reviews = :val1',

            ExpressionAttributeValues={
                ':val1' : review_count + 1,

            }
        )

        final_table.update_item(
            Key={
                'emp_id':get_employee_id
            },
            UpdateExpression='SET score = :val2',

            ExpressionAttributeValues={
                ':val2' : score + 15
            }
        )
    print("Updated suceesfully")


    return render_template('home.html',solu="Updated Successfully")


@app.route('/result')
def result():
    client = boto3.client('dynamodb',aws_access_key_id = access_key_id,aws_secret_access_key =
secret_access_key,aws_session_token = session_token,region_name = region )
    db = boto3.resource('dynamodb',aws_access_key_id = access_key_id,aws_secret_access_key =
secret_access_key,aws_session_token = session_token,region_name = region )







    table = db.Table('my_table2')
    id=['Emp_001','Emp_002','Emp_003','Emp_004','Emp_005']
    score=[]
    review=[]
    for i in id:
        resp = table.get_item(
                Key={
                    'emp_id' : i,
                }
```

```
          )
        d= (resp['Item']['score'])
        e=(resp['Item']['no_of_reviews'])
        d=int(d)
        e=int(e)
        score.append(d)
        review.append(e)
    print(score)
    print(review)
    return render_template("draw.html", review = review, score = score)


if __name__=='__main__':
    app.run(debug= True)
```

- HTML page

```
<html>
    <head>

        <title>Employee survey</title>
        <body bgcolor="powderblue">
           <center><h1 style="font-size:42px">CUSTOMER FEEDBACK ON EMPLOYEE</h1><br><br><br><br>
            <form action="http://127.0.0.1:5000/process"method="post","get">
 <div style="font-size:35px;background-color:wheat">
    <B><p>Customer Name:<input type="text" name="customer_name" required style="font-size:25px;"></p>

<p>
    <h3> Select employee:
        <select type="text" name="emp" required="required"  style="font-size:25px;">
        <option value="Emp_001">Emp_001</option>
        <option value="Emp_002">Emp_002</option>
        <option value="Emp_003">Emp_003</option>
        <option value="Emp_004">Emp_004</option>
         <option value="Emp_005">Emp_005</option>
        </h3></B>
      </select><br><br>
        Review:<textarea  name="review" rows="4" cols="50" style="font-size:24px;"></textarea>


</p>



<button type="submit" style="font-size:23px;color:black;" class="btn btn-primary btn-block
btn-large">submit</button>
            </form></div>
            <h2 style="color:green;">{{solu}}</h4></center>
        </body>
    </head>
</html>
```

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
    <script type="text/javascript">
      google.charts.load("current", {packages:["corechart"]});
      google.charts.setOnLoadCallback(drawChart);
      function drawChart() {
        var data = google.visualization.arrayToDataTable([
          ['Task', 'no_of_reviews'],
          ['Employee_1',     {{review[0]}}],
          ['Employee_2',      {{review[1]}}],
          ['Employee_3',  {{review[2]}}],
          ['Employee_4', {{review[3]}}],
          ['Employee_5',    {{review[4]}}]
        ]);

        var options = {
          title: 'No_of_reviews_got_from_customer',
          is3D: true,
        };

        var chart = new google.visualization.PieChart(document.getElementById('Employee_score'));
        chart.draw(data, options);
      }
    </script>


<script type="text/javascript">
    google.charts.load("current", {packages:["corechart"]});
    google.charts.setOnLoadCallback(drawChart);
    function drawChart() {
      var data = google.visualization.arrayToDataTable([
        ['Task', 'score'],
        ['Employee_1',    {{score[0]}}],
        ['Employee_2',       {{score[1]}}],
        ['Employee_3',   {{score[2]}}],
        ['Employee_4',  {{score[3]}}],
        ['Employee_5',     {{score[4]}}]
      ]);

      var options = {
        title: 'Score_for_the_employee',
        is3D: true,
      };

      var chart = new google.visualization.PieChart(document.getElementById('Employee_review_count'));
      chart.draw(data, options);
    }
  </script>
```

```
    <style>
        .chart
        {
            margin:20;
            display:flex;
        }
    </style>

  </head>
  <body bgcolor="#ffff99">
  <center><h1>EMPLOYEE SCORE REPORT</h1></center><BR><BR>
      <div class = "chart">
    <div id="Employee_score" style="width: 900px; height: 500px;"></div>
    <div id="Employee_review_count" style="width: 900px; height: 500px;"></div>
      </div>
  </body>
</html>
```