

TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

Faculty of Computing and Information Technology

EMPLOYEE MANAGEMENT SYSTEM

(Programmed Title and tutorial group)

Practical Assignment

BAIT3273 Cloud Computing (202301)

LECTURER: LOW CHOON KEAT

Name (Block Capital)	Registration No.
Liew Min Hui	21WMR08652
Ng Min Xiong	21WMR09435
Sit Yie Sian	21WMR03693

Grading Rubric for Practical Assignment (Form 2)

Levels of Assessment

Criteria/Marks Allocation		Excellent (9-10)	Good (6-8)	Average (4-5)	Poor 1-3	Comments by Lecturer (if necessary)	Score	
1.	Able to deploy resource group including server (with security features), database, web app and application plan.							
2.	Architecture Diagram							
3.	Attractive GUI Design							
4.	Organized Database design (table and field) and S3 Storage							
5.	Auto Scaling and Load Balancing							
6.	Proper snippet on report							
7.	Able to analyze, configure, troubleshoot and work independently with minimal guidance							
8.	Demonstration Session and Student Effort (30 marks)							
		STUDENT NAME			Comments by Lecturer		Score	TOTAL (100%)
		1.4.1 Liew Min Hui						
		2.4.1 Ng Min Xiong						
		3.4.1 Sit Yie Sian						

Acknowledgement

We would like to thank our cloud computing lecturer, Mr. Low Choon Keat, for guiding us throughout this project. His feedback and support helped us learn and improve our cloud computing skills.

We also appreciate the hard work and teamwork of our group members in completing this project.

Finally, we want to acknowledge the following cloud services used in this project: Amazon Simple Storage Service (Amazon S3), Amazon RDS, and Cloud Watch. These services helped us build an effective employee database system.

Thank you to everyone who supported us on this journey.

Table of content

Table of content	3
1.0 Introduction	4
2.0 AWS services used with screenshot of resources deployed	5
2.1 Amazon Elastic Compute Cloud (EC2)	5
2.2 Amazon Simple Storage Service (Amazon S3)	6
2.3 Amazon Relational Database Service (Amazon RDS)	8
2.4 Amazon CloudWatch	9
2.5 Amazon Auto Scaling	10
2.6 Elastic Load Balancing	11
3.0 Solution Architecture	12
4.0 Web Page	13
4.1 Web Page URL	13
4.2 Snippet of Web page(s)	13
4.2.1 Home page	13
4.2.2 Add Employee Data Page	14
4.2.3 Get Employee Information Page	15
4.2.4 Delete Employee Data Page	17
4.2.5 About us page	18
4.3 Database table, field, and records	19
4.4 Codes: SQL and GUI	21
4.4.1 EmpAPP.py	21
4.4.2 AboutUs.html	25
4.4.3 AddEmp.html	28
4.4.4 AddEmpOutput.html	31
4.4.5 DeleteEmp.html	32
4.4.6 DeleteEmpOutput.html	34
4.4.7 GetEmp.html	36
4.4.8 GetEmpOutput.html	39
4.4.9 home.html	41
5.0 Conclusion	43
6.0 Formatted Reference	44

1.0 Introduction

In today's fast-paced and highly competitive business environment, organizations are increasingly turning to cloud computing to streamline their operations and enhance their productivity. Cloud computing offers a scalable, flexible, and cost-effective way to store and manage data, run applications, and deliver services over the internet. This technology has become an essential tool for businesses of all sizes and across all industries.

The small startup company we worked with is planning to launch a new employee database management system using a LAMP stack composed of open-source software. With the expectation of significant growth in the near future, they are concerned about scalability, disaster recovery planning, data analytics, database performance, load distribution, and creating a self-healing infrastructure.

As part of our cloud computing course, we were tasked with building a system that includes a web page with an employee database, user-generated media files stored in Amazon Simple Storage Service (Amazon S3), a highly available and durable storage service, and an Amazon RDS SQL database for storing and querying employee data. Additionally, we utilized Cloud Watch to monitor resources and applications in real-time.

In this report, we will describe the development process of our employee database system, including the challenges we faced and the solutions we implemented. We will also provide a detailed analysis of the cloud services we utilized, including their features, benefits, and limitations. Our goal is to share our experience and knowledge in cloud computing with others who may be interested in this field.

2.0 AWS services used with screenshot of resources deployed

2.1 Amazon Elastic Compute Cloud (EC2)

ew EC2 Experience
All us what you think

Dashboard

Global View

its

ts

inces

ances

ince Types

ch Templates

t Requests

ngs Plans

rved Instances

Instances (6) Info

Find instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
<input type="checkbox"/>	assignment	i-06ee9749027da91d3	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-23-20-195-8.comp...	23.20.195.8	-
<input type="checkbox"/>	-	i-03b49435d8f54d07f	Terminated	t2.micro	-	No alarms	us-east-1a	-	-	-
<input type="checkbox"/>	-	i-08804652a29e2271a	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-18-206-209-30.co...	18.206.209.30	-
<input type="checkbox"/>	-	i-0e25908bf472709c2	Terminated	t2.micro	-	No alarms	us-east-1b	-	-	-
<input type="checkbox"/>	-	i-02e15722ac36dd5fc	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-35-168-15-132.co...	35.168.15.132	-
<input type="checkbox"/>	-	i-015207a69f3b5e1ea	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	ec2-44-200-44-146.co...	44.200.44.146	-

Instance: i-06ee9749027da91d3 (assignment)

Auto-assigned IP address
23.20.195.8 [Public IP]

IAM Role
LabRole

IMDSv2
Required

Instance details Info

Platform
Amazon Linux (Inferred)

Platform details
Linux/UNIX

Stop protection

VPC ID
vpc-04b57005dd05cfa97

Subnet ID
subnet-06583d40b10bddb21

AMI ID
ami-0889a44b331db0194

AMI name
al2023-ami-2023.0.20230503.0-kernel-6.1-x86_64

Launch time

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer f

Auto Scaling Group name
-

Monitoring
disabled

Termination protection
Disabled

AMI location

2.2 Amazon Simple Storage Service (Amazon S3)

Amazon S3 > Buckets > liewminhui-employee

liewminhui-employee [Info](#)

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (0)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

< 1 >

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
No objects					
You don't have any objects in this bucket.					

Upload

liewminhui-employee [Info](#)

Objects | Properties | Permissions | Metrics | Management | Access Points

Objects (5)
Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need

Copy S3 URI Copy URL Download Open Delete Actions Create folder

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	emp-id-1_image_file	-	May 8, 2023, 17:33:38 (UTC+08:00)
<input type="checkbox"/>	emp-id-2_image_file	-	May 8, 2023, 17:34:24 (UTC+08:00)
<input type="checkbox"/>	emp-id-3_image_file	-	May 8, 2023, 17:34:45 (UTC+08:00)
<input type="checkbox"/>	emp-id-4_image_file	-	May 8, 2023, 17:35:15 (UTC+08:00)
<input type="checkbox"/>	emp-id-5_image_file	-	May 8, 2023, 17:35:48 (UTC+08:00)

liewminhui-s3-bucket [Info](#)

Publicly accessible

Objects

Properties

Permissions

Metrics


Management





Access Points

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll r

  Copy S3 URI  Copy URL  Download  Open  Delete  Actions  Create folder

 Find objects by prefix

<input type="checkbox"/>	Name	▲	Type	▼	Last modified
<input type="checkbox"/>	 1.jpg		jpg		May 7, 2023, 22:15:28 (UTC+08:00)
<input type="checkbox"/>	 123.png		png		May 7, 2023, 22:15:27 (UTC+08:00)
<input type="checkbox"/>	 2.jpg		jpg		May 7, 2023, 22:15:30 (UTC+08:00)
<input type="checkbox"/>	 3.jpg		jpg		May 7, 2023, 22:15:21 (UTC+08:00)

2.3 Amazon Relational Database Service (Amazon RDS)

RDS > Databases > employee

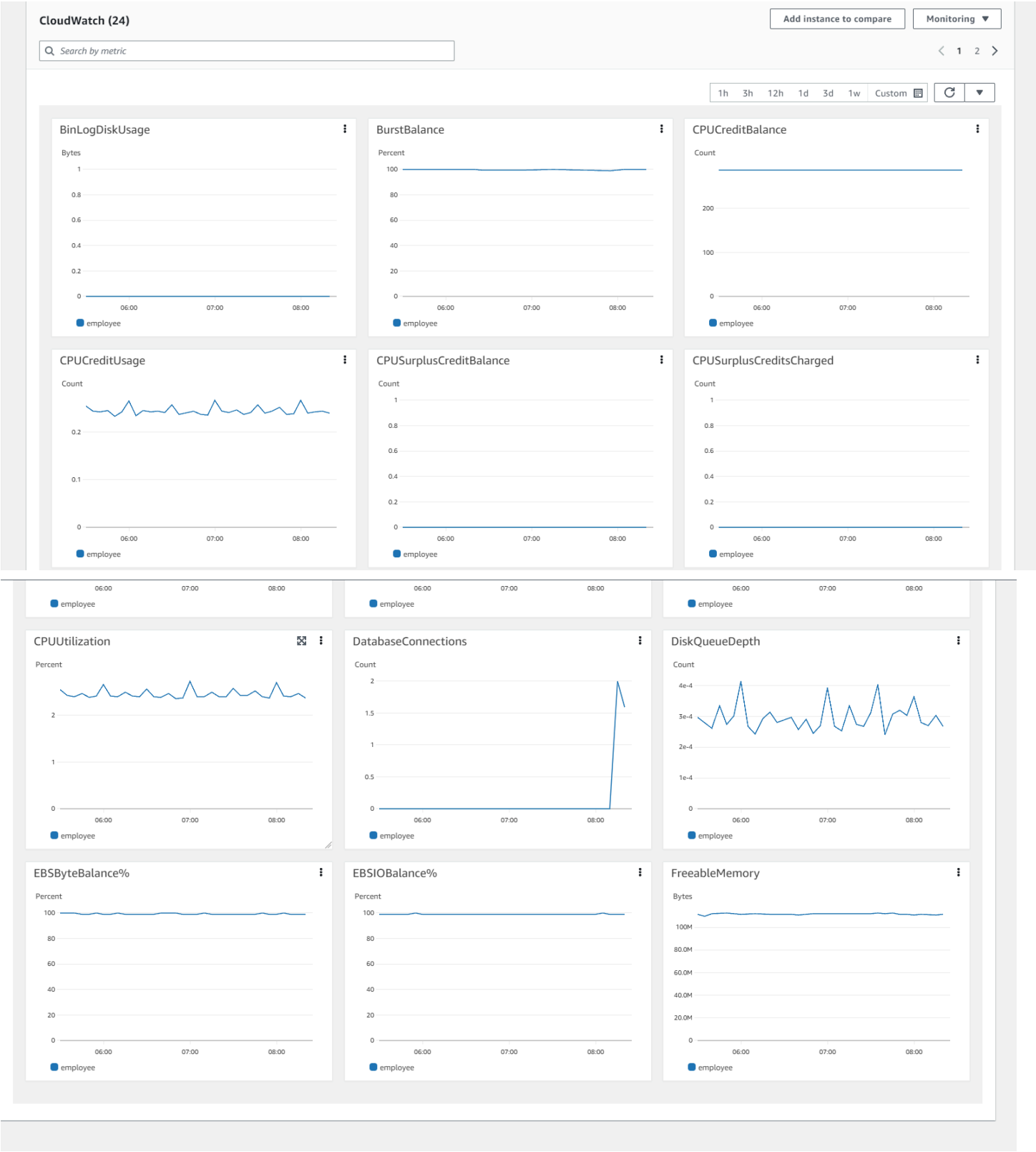
employee

Modify

Summary

DB identifier employee	CPU <div>2.68%</div>	Status Available	Class db.t3.micro
Role Instance	Current activity <div>0 Connections</div>	Engine MySQL Community	Region & AZ us-east-1b

2.4 Amazon CloudWatch



2.5 Amazon Auto Scaling

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Scheduled Instances

Capacity Reservations

▼ Images

AMIs

AMI Catalog

▼ Elastic Block Store

Volumes

Snapshots

Lifecycle Manager

▼ Network & Security

Security Groups

Elastic IPs

Placement Groups

Key Pairs

Network Interfaces

▼ Load Balancing

Load Balancers

Target Groups

▼ Auto Scaling

Dynamic scaling policy created or edited successfully.

EC2 > Auto Scaling groups > AWS1

AWS1

Details | Activity | Automatic scaling | Instance management | Monitoring | Instance refresh

Scaling policies resize your Auto Scaling group to meet changes in demand. With reactive dynamic scaling policies, you can track specific predictive scaling policies along with dynamic scaling policies in the following situations: when your application demand changes quickly,

Dynamic scaling policies (1) [Info](#)

Target Tracking Policy

Target tracking scaling

Enabled

As required to maintain Average CPU utilization at 55

Add or remove capacity units as required

300 seconds to warm up before including in metric

Enabled

Auto Scaling group updated successfully

EC2 > Auto Scaling groups > AWS1

AWS1

Details | Activity | Automatic scaling | Instance management | Monitoring | Instance refresh

Group details

Auto Scaling group name

AWS1

Date created

Sun May 07 2023 15:30:51 GMT+0800 (Malaysia Time)

Desired capacity

3

Minimum capacity

2

Maximum capacity

5

Status

Updating capacity

Amazon Resource Name (ARN)

arn:aws:autoscaling:us-east-1:216224819496:autoScalingGroup:a519cff3-4d1e-47d1-9b9b-3474e94c1698:autoScalingGroupName/AWS1

Edit

Launch template

Edit

2.6 Elastic Load Balancing

ELB / Load balancers / LB1

LB1

Refresh

Actions

▼ Details

arn:aws:elasticloadbalancing:us-east-1:216224819496:loadbalancer/app/LB1/eb49867ed642a355

Load balancer type Application	DNS name LB1-785444116.us-east-1.elb.amazonaws.com (A Record)	Status Active	VPC vpc-04b57005dd05cfa97
IP address type IPv4	Scheme Internet-facing	Availability Zones subnet-06583d40b10bddb21 us-east-1a (use1-az4) subnet-007b665fa3e0a224b us-east-1b (use1-az6) subnet-008b0b79879e46a2c us-east-1c (use1-az1)	Hosted zone Z35SXDOTRQ7X7K
Date created May 7, 2023, 16:04 (UTC+08:00)			

Listeners

Network mapping

Security

Monitoring

Integrations

Attributes

Tags

Listeners (1)

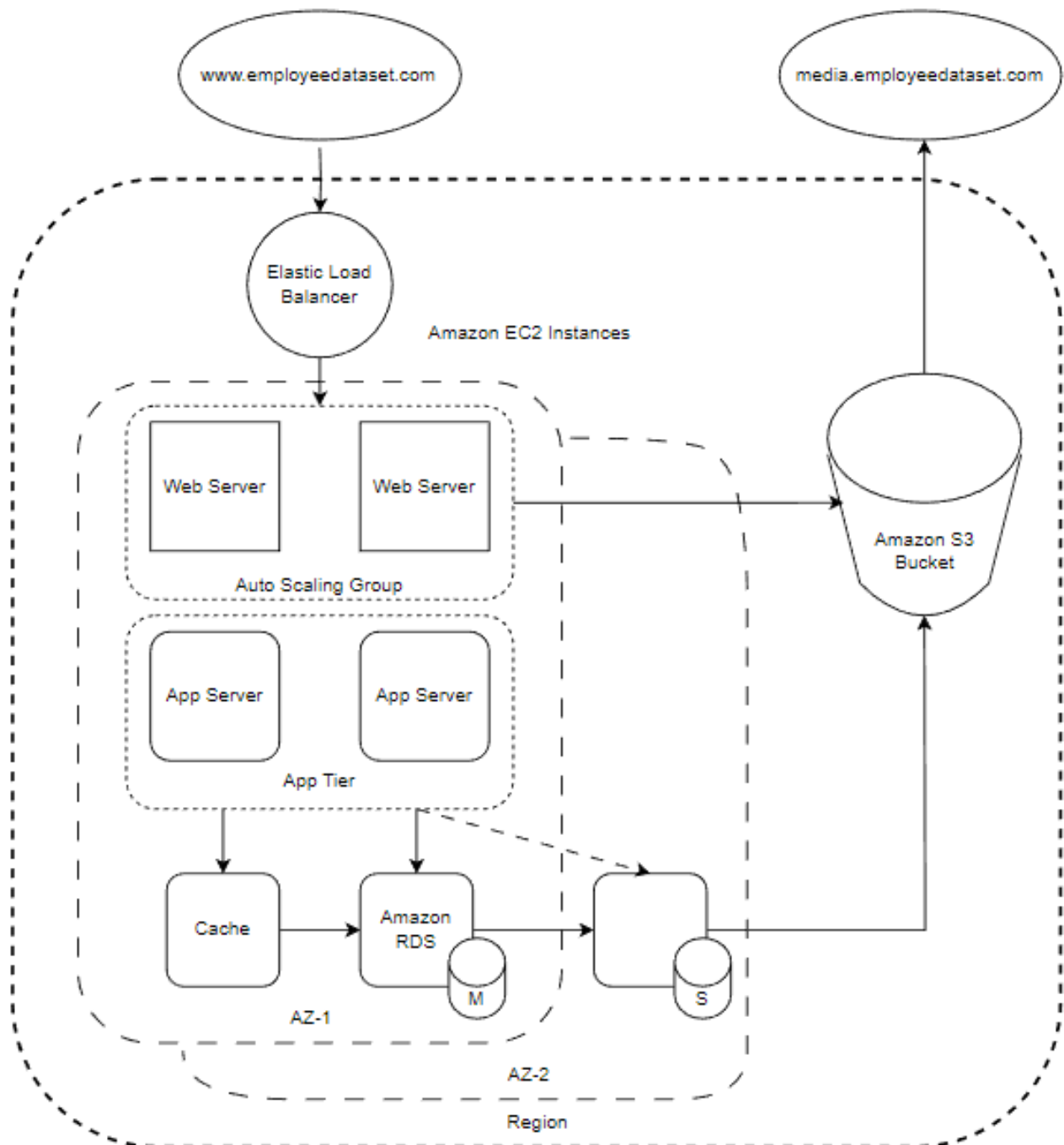
A listener checks for connection requests on its port and protocol. Traffic received by the listener is routed according to its rules.

Search

< 1 > ⌕

	Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL cert	Tags
<input type="checkbox"/>	HTTP:80	<div>Forward to target group<ul style="list-style-type: none">TG1: 1 (100%)Group-level stickiness: Off</div>	1 rule	ARN	Not applicable	Not applicable	0 tags

3.0 Solution Architecture



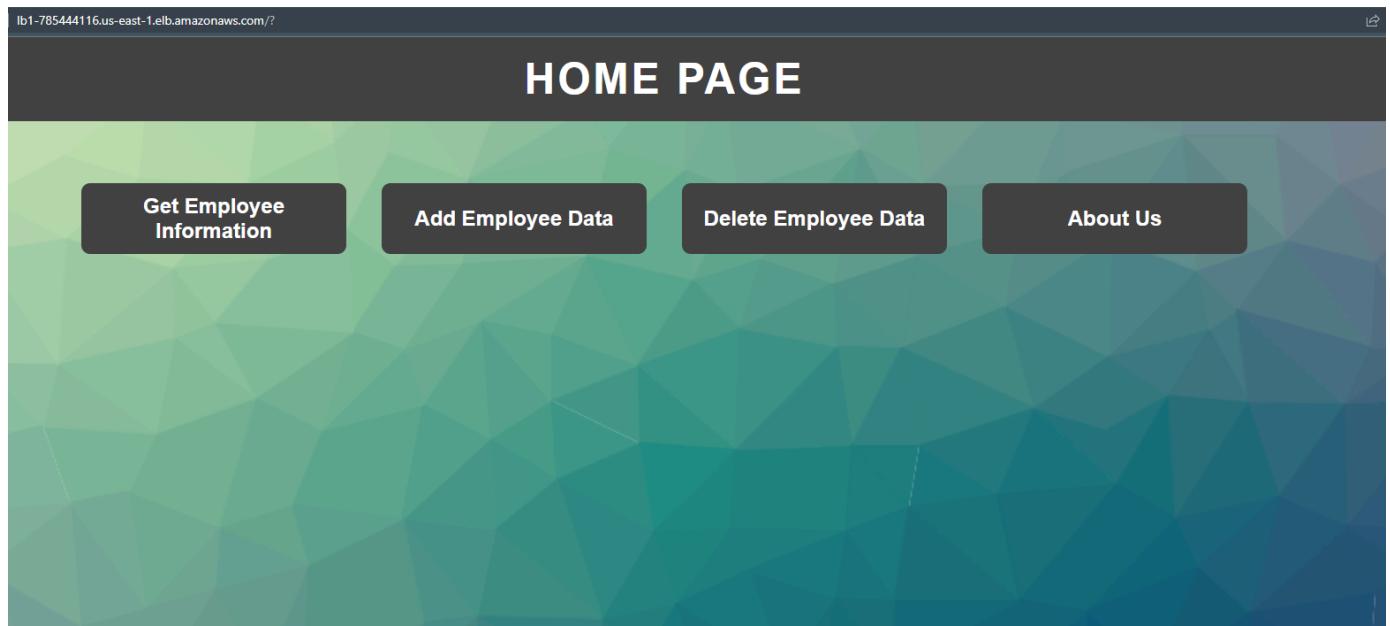
4.0 Web Page

4.1 Web Page URL

<http://lb1-785444116.us-east-1.elb.amazonaws.com/>

4.2 Snippet of Web page(s)

4.2.1 Home page



4.2.2 Add Employee Data Page

lb1-785444116.us-east-1.elb.amazonaws.com/add?

Employee Database

Hire Date:

Profile Picture:
 No file chosen

lb1-785444116.us-east-1.elb.amazonaws.com/addemp

Save Successful

Following Employee has been added to the database
yiesian sit

4.2.3 Get Employee Information Page

lb1-785444116.us-east-1.elb.amazonaws.com/get?

Employee Information

Enter Employee ID

Fetch Info

Go to Home Page

Employee Information

Employee ID:

2

First Name:

yiesian

Last Name:

sit

Primary Skill:

Cloud

Location:

selangor

Payscale:

9999

Hire Date:

2023-05-08

[Back](#)

4.2.4 Delete Employee Data Page

lb1-785444116.us-east-1.elb.amazonaws.com/delete?

Delete Employee

Employee ID:

Delete

Go Back

lb1-785444116.us-east-1.elb.amazonaws.com/deleteemp

Delete Employee

Employee with ID 2 has been deleted.

Back to Home

4.2.5 About us page

lb1-785444116.us-east-1.elb.amazonaws.com/about?

ABOUT US



LALA

As a student in cloud courses, I'm passionate about learning more about cloud computing and its potential to solve real-world problems.



MIN HUI

Hi, I'm Liew Min Hui. I have a strong passion for technology and enjoy exploring the latest advancements in the field. My passion for learning and exploring new technologies drives me to seek out new challenges and opportunities to expand my knowledge.



YIE SIEN

Sit Yie Sian is a data analyst with a background in quantitative analysis and operations management. Skilled in data modeling and analysis tools, and experienced in leading teams to implement process improvements.

4.3 Database table, field, and records

Table Name	Field Name	Field Type
Employee	emp_id	TEXT
	first_name	TEXT
	last_name	TEXT
	pri_skill	TEXT
	location	TEXT
	payscale	TEXT
	hire_date	DATE

Host: employee.cmoy300s7jni.us-east-1.rds.amazonaws.com Database: employee Table: employee Data

Basic Options Indexes (0) Foreign keys (0) Check constraints (0) Partitions CREATE code

Name: employee

Comment:

Columns: + Add × Remove ▲ Up ▼ Down

#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default	Co
1	emp_id	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
2	first_name	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
3	last_name	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
4	pri_skill	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
5	location	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
6	payscale	TEXT		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	
7	hire_date	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL	

emp_id	first_name	last_name	pri_skill	location	payscale	hire_date
1	Min Hui	Liew	Cloud	Perak	5000	2022-05-04
2	Min Xiong	Ng	Cloud	selangor	4000	2022-07-11
3	yiesian	sit	Cloud	selangor	9999	2022-08-23
4	wenjie	Ng	Cloud	Kuantan	4000	2022-05-07
5	Waileong	Teng	Cloud	selangor	5000	2022-01-17

employee.employee: 5 rows total (approx) >> Next  Show all |  Sorting (4) 

emp_id ▼ ₄	first_name	last_name ▼ ₃	pri_skill ▼ ₂	location ▼ ₁	payscale	hire_date
5	Waileong	Teng	Cloud	selangor	5000	2022-01-17
3	yiesian	sit	Cloud	selangor	9999	2022-08-23
2	Min Xiong	Ng	Cloud	selangor	4000	2022-07-11
1	Min Hui	Liew	Cloud	Perak	5000	2022-05-04
4	wenjie	Ng	Cloud	Kuantan	4000	2022-05-07

4.4 Codes: SQL and GUI

4.4.1 EmpAPP.py

```
from flask import Flask, render_template, request
from pymysql import connections
import os
import boto3
from config import *

app = Flask(__name__)

bucket = custombucket
region = customregion

db_conn = connections.Connection(
    host=customhost,
    port=3306,
    user=customuser,
    password=custompass,
    db=customdb
)

output = {}
table = 'employee'

@app.route("/", methods=['GET', 'POST'])
def home():
    return render_template('home.html')

@app.route("/add", methods=['GET', 'POST'])
def add():
    return render_template('AddEmp.html')

@app.route("/delete", methods=['GET', 'POST'])
def delete():
    return render_template('DeleteEmp.html')

@app.route("/about", methods=['GET', 'POST'])
def about():
    return render_template('AboutUs.html')

@app.route("/get", methods=['GET', 'POST'])
def get():
    return render_template('GetEmp.html')
```

```

@app.route("/addemp", methods=['POST'])
def AddEmp():
    emp_id = request.form['emp_id']
    first_name = request.form['first_name']
    last_name = request.form['last_name']
    pri_skill = request.form['pri_skill']
    location = request.form['location']
    payscale = request.form['payscale']
    hire_date = request.form['hire_date']
    emp_image_file = request.files['emp_image_file']

    insert_sql = "INSERT INTO employee VALUES (%s, %s, %s, %s, %s,
%s, %s)"
    cursor = db_conn.cursor()

    if emp_image_file.filename == "":
        return "Please select a file"

    try:
        cursor.execute(insert_sql, (emp_id, first_name, last_name,
pri_skill, location, payscale, hire_date))
        db_conn.commit()
        emp_name = "" + first_name + " " + last_name
        # Uplaod image file in S3 #
        emp_image_file_name_in_s3 = "emp-id-" + str(emp_id) +
"_image_file"
        s3 = boto3.resource('s3')

        try:
            print("Data inserted in MySQL RDS... uploading image to
S3...")

            s3.Bucket(custombucket).put_object(Key=emp_image_file_name_in_s3,
Body=emp_image_file)
            bucket_location =
boto3.client('s3').get_bucket_location(Bucket=custombucket)
            s3_location = (bucket_location['LocationConstraint'])

            if s3_location is None:
                s3_location = ''
            else:
                s3_location = '-' + s3_location

            object_url =
"https://s3{0}.amazonaws.com/{1}/{2}".format(
                s3_location,
                custombucket,
                emp_image_file_name_in_s3)

        except Exception as e:
            return str(e)

```

```

        finally:
            cursor.close()

        print("all modification done...")
        return render_template('AddEmpOutput.html', name=emp_name)

# Define the fetchdata route
@app.route('/fetchdata', methods=['POST'])
def fetchdata():
    emp_id = request.form['emp_id']
    cursor = db_conn.cursor()
    cursor.execute("SELECT * FROM employee WHERE emp_id = %s",
(emp_id,))
    result = cursor.fetchone()
    cursor.close()
    if result is not None:
        emp_data = {
            'id': result[0],
            'fname': result[1],
            'lname': result[2],
            'pri_skill': result[3],
            'location': result[4],
            'payscale': result[5],
            'hiredate': result[6]
        }
        return render_template('GetEmpOutput.html', **emp_data)
    else:
        return "Employee not found"

@app.route("/deleteemp", methods=['POST'])
def DeleteEmp():
    emp_id = request.form['emp_id']
    cursor = db_conn.cursor()

    try:
        # Delete image file from S3 bucket
        emp_image_file_name_in_s3 = "emp-id-" + str(emp_id) +
"_image_file"
        s3 = boto3.resource('s3')
        s3.Object(custombucket, emp_image_file_name_in_s3).delete()

        # Delete employee from database
        cursor.execute("DELETE FROM employee WHERE emp_id = %s",
(emp_id,))
        db_conn.commit()

    except Exception as e:
        return str(e)

    finally:

```



```
        cursor.close()

    print("all modification done...")
    return render_template('DeleteEmpOutput.html', id=emp_id)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=80, debug=True)
```

4.4.2 AboutUs.html

```
<!DOCTYPE html>
<html>
<head>
  <title>About Us</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      margin: 0;
      padding: 0;
    }

    header {
      background-color: #f5f5f5;
      padding: 20px;
      text-align: center;
    }

    h1 {
      margin: 0;
      font-size: 48px;
      font-weight: bold;
      color: #444;
      text-transform: uppercase;
      letter-spacing: 3px;
    }

    .container {
      max-width: 1500px;
      margin: 0 auto;
      padding: 50px;
      display: flex;
      flex-wrap: wrap;
      justify-content: center;
    }

    .member {
      background-color: #f5f5f5;
      margin: 20px;
      width: calc(33.33% - 40px);
      box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
      transition: box-shadow 0.3s ease-in-out;
    }

    .member:hover {
      box-shadow: 0px 0px 20px rgba(0, 0, 0, 0.2);
    }
  </style>
</head>
</html>
```

```

    }

    .member img {
        width: 100%;
        height: 500px;
        object-fit: cover;
    }

    .member h3 {
        margin: 20px 0 10px 0;
        font-size: 24px;
        font-weight: bold;
        color: #444;
        text-transform: uppercase;
        letter-spacing: 2px;
    }

    .member p {
        margin: 0;
        font-size: 16px;
        line-height: 1.5;
        color: #666;
        text-align: justify;
    }

    @media screen and (max-width: 768px) {
        .member {
            width: calc(50% - 40px);
        }
    }

    @media screen and (max-width: 480px) {
        header {
            padding: 10px;
        }

        h1 {
            font-size: 36px;
        }

        .member {
            width: calc(100% - 40px);
        }
    }
</style>
</head>
<body>
    <header>
        <h1>About Us</h1>
    </header>

    <div class="container">

```

```

        <div class="member">
            
            <h3>LaLa</h3>
            <p>As a student in cloud courses, I'm passionate about
learning more about cloud computing and its potential to solve
real-world problems.</p>
        </div>

        <div class="member">
            
            <h3>Min Hui</h3>
            <p>Hi, I'm Liew Min Hui. I have a strong passion for
technology and enjoy exploring the latest advancements in the field.
My passion for learning and exploring new technologies drives me to
seek out new challenges and opportunities to expand my knowledge.</p>
        </div>

        <div class="member">
            
            <h3>Yie Sien</h3>
            <p>Sit Yie Sian is a data analyst with a background in
quantitative analysis and operations management. Skilled in data
modeling and analysis tools, and experienced in leading teams to
implement process improvements.</p>
        </div>
        <form action="/" autocomplete="on" method="GET">
            <button type="submit" style="display: block; text-align:
center; font-size: 24px; font-weight: bold; color: #fff;
background-color: #444; padding: 10px 20px; border-radius: 10px;
text-decoration: none; margin: 50px auto; border:
none;">Back</button>
        </form>
    </div>

</body>

</html>

```

4.4.3 AddEmp.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Employee Database</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
    }
    .container {
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
      box-sizing: border-box;
      background-color: rgba(255, 255, 255, 0.9);
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
    }
    .header {
      background-color: #343a40;
      color: #fff;
      padding: 10px;
      text-align: center;
      border-radius: 10px 10px 0 0;
    }
    h1 {
      font-size: 36px;
      margin: 0;
    }
    form {
      padding: 20px;
      border-radius: 0 0 10px 10px;
    }
    input[type="number"],
    input[type="text"],
    input[type="file"] {
      display: block;
      width: 100%;
      padding: 10px;
      font-size: 16px;
      margin-bottom: 20px;
      border-radius: 5px;
      border: none;
      box-shadow: 0 0 5px rgba(0,0,0,.2);
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="header">
      <h1>Employee Database</h1>
    </div>
    <form>
      <input type="text" value="Name" />
      <input type="text" value="Email" />
      <input type="text" value="Phone" />
      <input type="text" value="Address" />
      <input type="text" value="City" />
      <input type="text" value="State" />
      <input type="text" value="Zip" />
      <input type="text" value="Password" />
      <input type="text" value="Confirm Password" />
      <input type="button" value="Add Employee" />
    </form>
  </div>
</body>
</html>
```

```

    }
    button[type="submit"] {
        background-color: #343a40;
        color: #fff;
        padding: 10px 20px;
        font-size: 18px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        transition: background-color .3s;
    }
    button[type="submit"]:hover {
        background-color: #23272b;
    }
    .link {
        display: inline-block;
        background-color: #17a2b8;
        color: #fff;
        padding: 10px 20px;
        font-size: 18px;
        text-decoration: none;
        border-radius: 5px;
        transition: background-color .3s;
    }
    .link:hover {
        background-color: #117a8b;
    }
    .button {
        display: inline-block;
        background-color: #343a40;
        color: #fff;
        padding: 10px 20px;
        font-size: 18px;
        text-decoration: none;
        border-radius: 5px;
        margin-bottom: 20px;
        cursor: pointer;
        transition: background-color .3s;
    }
    .button:hover {
        background-color: #23272b;
    }
    .footer {
        text-align: center;
        font-size: 18px;
    }
}
</style>
</head>
<body>
    <div class="container">
        <div class="header">
            <h1>Employee Database</h1>

```

```

        </div>
        <form action="/addemp" autocomplete="on" method="POST"
enctype="multipart/form-data">
            <input type="text" name="emp_id" placeholder="Employee
ID" autofocus>
            <input type="text" name="first_name" placeholder="First
Name">
            <input type="text" name="last_name" placeholder="Last
Name">
            <input type="text" name="pri_skill" placeholder="Primary
Skills">
            <input type="text" name="location"
placeholder="Location">
            <input type="text" name="payscale"
placeholder="Payscale">
            <label for="hire_date">Hire Date:</label>
            <input type="date" name="hire_date"
id="hire_date"><br><br>
            <label for="emp_image_file">Profile Picture:</label>
            <input type="file" name="emp_image_file"
accept="image/*">
            <button type="submit">UPDATE DATABASE</button>
        </form>
        <div class="footer">
            <form action = "/" autocomplete="on" method = "GET">
                <button type = "submit">Go To Home Page</button>
            </form>
        </div>
    </div>
</body>
</html>

```

4.4.4 AddEmpOutput.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Save Successful</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      color: #fff;
    }
    .container {
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
      box-sizing: border-box;
      background-color: rgba(255, 255, 255, 0.9);
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
      margin-top: 50px;
    }
    h1 {
      font-size: 36px;
      margin: 0;
      color: DodgerBlue;
      text-align: center;
    }
    h2 {
      font-size: 24px;
      margin: 0;
      text-align: center;
      color: black;
    }
    button[type="submit"] {
      background-color: #343a40;
      color: #fff;
      padding: 10px 20px;
      font-size: 18px;
      border: none;
      border-radius: 5px;
      cursor: pointer;
      transition: background-color .3s;
      margin-top: 20px;
    }
    button[type="submit"]:hover {
```



```

        background-color: #23272b;
    }
</style>
</head>
<body>
    <div class="container">
        <h1>Save Successful</h1>
        <h2>Following Employee has been added to the
database</h2>
        <h2>{{ name }}</h2>
        <form action="/add" autocomplete="on" method = "GET">
            <button type="submit">Go Back</button>
        </form>
    </div>
</body>
</html>

```

4.4.5 DeleteEmp.html

```

<!DOCTYPE html>
<html>
<head>
    <title>Delete Employee</title>
    <style>
        body {
            background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
            background-size: cover;
            background-repeat: no-repeat;
            background-attachment: fixed;
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
        }
        h1 {
            color: dodgerblue;
            text-align: center;
            font-size: 36px;
            margin-top: 50px;
        }
        form {
            margin: auto;
            max-width: 800px;
            padding: 20px;
            box-sizing: border-box;
            background-color: rgba(255, 255, 255, 0.9);
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
        }
        label {

```

```

        display: block;
        font-size: 20px;
        font-weight: bold;
        margin-top: 20px;
    }
    input[type="text"], select {
        padding: 10px;
        font-size: 16px;
        border: none;
        border-radius: 5px;
        background-color: #f5f5f5;
        margin-top: 10px;
        width: 100%;
    }
    button[type="submit"] {
        background-color: grey;
        color: white;
        padding: 10px 20px;
        font-size: 20px;
        border: none;
        border-radius: 5px;
        cursor: pointer;
        font-style: oblique;
        margin-top: 20px;
        transition: background-color .3s;
    }
    button[type="submit"]:hover {
        background-color: #444;
    }
</style>
</head>
<body>
    <h1>Delete Employee</h1>
    <form method="POST" action="/deleteemp">
        <label for="emp_id">Employee ID:</label>
        <input type="text" id="emp_id" name="emp_id" required>

        <button type="submit">Delete</button>
    </form>
    <form action="/" autocomplete="on" method = "GET">
        <button type="submit">Go Back</button>
    </form>
</body>
</html>

```

4.4.6 DeleteEmpOutput.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Delete Employee</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
    }
    h1 {
      color: dodgerblue;
      text-align: center;
      font-size: 36px;
      margin-top: 50px;
    }
    form {
      margin: auto;
      max-width: 800px;
      padding: 20px;
      box-sizing: border-box;
      background-color: rgba(255, 255, 255, 0.9);
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
    }
    label {
      display: block;
      font-size: 20px;
      font-weight: bold;
      margin-top: 20px;
    }
    button[type="submit"] {
      background-color: grey;
      color: white;
      padding: 10px 20px;
      font-size: 20px;
      border: none;
      border-radius: 5px;
      cursor: pointer;
      font-style: oblique;
      margin-top: 20px;
      transition: background-color .3s;
    }
    button[type="submit"]:hover {
      background-color: #444;
    }
  </style>
</head>
<body>
  <h1>Delete Employee</h1>
  <form>
    <label>Delete Employee</label>
    <button type="submit">Delete Employee</button>
  </form>
</body>
</html>
```

```
        }
    </style>
</head>
<body>
    <h1>Delete Employee</h1>
    <form action="/delete" autocomplete="on" method = "GET">
        <p>Employee with ID {{ id }} has been deleted.</p>
        <button type="submit" >Back to Home</button>
    </form>
</body>
</html>
```

4.4.7 GetEmp.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Get Employee Information</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-repeat: no-repeat;
      background-size: cover;
      background-position: center center;
      background-attachment: fixed;
      font-family: 'Arial', sans-serif;
    }

    .container {
      max-width: 600px;
      margin: 0 auto;
      text-align: center;
      padding-top: 50px;
    }

    h1 {
      color: #1c69b5;
      font-weight: bold;
      font-size: 36px;
      margin-bottom: 30px;
    }

    h2 {
      color: #fff;
      font-size: 20px;
      margin-bottom: 20px;
      text-shadow: 0px 0px 5px rgba(0,0,0,0.5);
    }

    input[type="text"] {
      width: 100%;
      padding: 10px;
      font-size: 18px;
      border: none;
      background-color: #fff;
      border-radius: 5px;
      box-shadow: 0 0 5px rgba(0, 0, 0, 0.2);
      margin-bottom: 20px;
    }

    button {
      padding: 10px 30px;
      font-size: 18px;
```

```

        color: #fff;
        border: none;
        background-color: #1c69b5;
        border-radius: 5px;
        box-shadow: 0 0 5px rgba(0, 0, 0, 0.2);
        cursor: pointer;
        margin-bottom: 20px;
    }

    button:hover {
        background-color: #145c9e;
    }

    .link {
        display: inline-block;
        background-color: #17a2b8;
        color: #fff;
        padding: 10px 20px;
        font-size: 18px;
        text-decoration: none;
        border-radius: 5px;
        transition: background-color .3s;
    }

    .link:hover {
        background-color: #117a8b;
    }

    a {
        color: #fff;
        font-size: 16px;
        text-decoration: none;
        text-shadow: 0px 0px 5px rgba(0,0,0,0.5);
    }

    a:hover {
        color: #fff;
    }
</style>
</head>
<body>
    <div class="container">
        <h1>Employee Information</h1>
        <form action="/fetchdata" autocomplete="on" method="POST">
            <h2>Enter Employee ID</h2>
            <input type="text" name="emp_id" placeholder="Employee
ID">
            <button type="submit">Fetch Info</button>
        </form>
        <form action = "/" autocomplete="on" method = "GET">
            <button type = "submit">Go to Home Page</button>
        </form>
    </div>

```

```
</body>  
</html>
```

4.4.8 GetEmpOutput.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Employee Information</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
    }
    h1 {
      color: dodgerblue;
      text-align: center;
      font-size: 36px;
      margin-top: 50px;
    }
    form {
      margin: auto;
      max-width: 800px;
      padding: 20px;
      box-sizing: border-box;
      background-color: rgba(255, 255, 255, 0.9);
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
    }
    label {
      display: block;
      font-size: 20px;
      font-weight: bold;
      margin-top: 20px;
    }
    p {
      font-size: 20px;
      margin-top: 10px;
    }
    button[type="submit"] {
      background-color: grey;
      color: white;
      padding: 10px 20px;
      font-size: 20px;
      border: none;
      border-radius: 5px;
      cursor: pointer;
      font-style: oblique;
      margin-top: 20px;
    }
  </style>
</head>
<body>
  <h1>Employee Information</h1>
  <form>
    <label>Name</label>
    <input type="text">
    <label>Age</label>
    <input type="text">
    <label>Gender</label>
    <input type="text">
    <label>Address</label>
    <input type="text">
    <label>City</label>
    <input type="text">
    <label>State</label>
    <input type="text">
    <label>Zip</label>
    <input type="text">
    <button type="submit">Submit</button>
  </form>
</body>
</html>
```



```

        transition: background-color .3s;
    }
    button[type="submit"]:hover {
        background-color: #444;
    }
</style>
</head>
<body>
    <h1>Employee Information</h1>
    <form>
        <label for="emp_id">Employee ID:</label>
        <p>{{ id }}</p>
        <label for="first_name">First Name:</label>
        <p>{{ fname }}</p>

        <label for="last_name">Last Name:</label>
        <p>{{ lname }}</p>

        <label for="pri_skill">Primary Skill:</label>
        <p>{{ pri_skill }}</p>

        <label for="location">Location:</label>
        <p>{{ location }}</p>

        <label for="payscale">Payscale:</label>
        <p>{{ payscale }}</p>

        <label for="hire_date">Hire Date:</label>
        <p>{{ hiredate }}</p>
    </form>
    <form action = "/get" autocomplete="on" method = "GET">
        <button type = "submit">Back</button>
    </form>
</body>
</html>

```

4.4.9 home.html

```
<!DOCTYPE html>
<html>
<head>
  <title>My Company</title>
  <style>
    body {
      background-image:
url("https://liewminhui-s3-bucket.s3.amazonaws.com/123.png");
      background-size: cover;
      background-repeat: no-repeat;
      background-attachment: fixed;
      font-family: Arial, sans-serif;
      background-color: #f5f5f5;
      margin: 0;
      padding: 0;
    }

    header {
      background-color: #444;
      padding: 20px;
      text-align: center;
      color: #fff;
    }

    h1 {
      margin: 0;
      font-size: 48px;
      font-weight: bold;
      text-transform: uppercase;
      letter-spacing: 3px;
    }

    main {
      max-width: 1500px;
      margin: 0 auto;
      padding: 50px;
      display: flex;
      flex-wrap: wrap;
      justify-content: center;
    }

    button {
      display: block;
      width: 300px;
      height: 80px;
      margin: 20px;
      font-size: 24px;
      font-weight: bold;
      color: #fff;
      background-color: #444;
```

```

        border: none;
        border-radius: 10px;
        cursor: pointer;
        transition: background-color 0.3s ease-in-out;
    }

    button:hover {
        background-color: #666;
    }
</style>
</head>
<body>
    <header>
        <h1>Home Page</h1>
    </header>

    <main>
        <form action="/get" autocomplete="on" method = "GET">
            <button type="submit">Get Employee Information</button>
        </form>
        <form action="/add" autocomplete="on" method = "GET">
            <button type="submit">Add Employee Data</button>
        </form>
        <form action="/delete" autocomplete="on" method = "GET">
            <button type="submit">Delete Employee Data</button>
        </form>
        <form action="/about" autocomplete="on" method = "GET">
            <button type="submit">About Us</button>
        </form>
    </main>
</body>
</html>

```

5.0 Conclusion

In conclusion, the small startup company's plan to launch a new employee database management system using a LAMP stack is an exciting opportunity for them to grow their business. However, they face several challenges related to scalability, disaster recovery planning, data analytics, database performance, load distribution, and creating a self-healing infrastructure. Our proposed solutions to these challenges include utilizing cloud-based solutions for scalability, implementing a robust disaster recovery plan, using data analytics tools for data insights, optimizing database performance, implementing load balancing techniques, and utilizing automation for self-healing infrastructure.

We hope that our recommendations will be useful for the small startup company in improving their employee database management system and achieving their goals. As cloud computing continues to grow in popularity, we believe that the skills and knowledge gained from this project will be valuable for anyone interested in similar challenges related to maintaining and scaling cloud-based systems.

6.0 Formatted Reference

1. ConceptDraw, (2023). 3-Tier Auto-scalable Web Application Architecture. Available at: <https://www.conceptdraw.com/examples/3-tier> Accessed at: 5/5/2023