

AWS Solutions Architect Course-end Project 1

Problem Statement :: Configure and Connect a MySQL Database Instance with a Web Server and Set up the Monitoring of the Solution

Description:

Your organization wants to deploy a new multi-tier application. The application will take live inputs from the employees, and it will be hosted on a web server running on the AWS cloud. The development team has asked you to set up the web server and configure it to scale automatically in cases of a traffic surge, to make the application highly available. They have also asked you to take the inputs from the employees and store them securely in the database.

Real-World Scenario:

Your organization wants to deploy a new multi-tier application. The application will take live inputs from the employees, and it will be hosted on a web server running on the AWS cloud. The development team has asked you to set up the web server and configure it to scale automatically in cases of a traffic surge, to make the application highly available. They have also asked you to take the inputs from the employees and store them securely in the database.

Skills used in the project and their usage in the industry are given below:

- AWS console - The AWS Management Console is a web application that includes and references several service consoles for managing AWS services.
- EC2 Instance - Amazon EC2 provides a large set of instance types that are customized to certain use cases.
- Apache Web Server - As a Web server, Apache is in charge of accepting directory (HTTP) requests from Internet users and delivering the requested data in the form of files and Web pages.

Expected Output:

Not secure | 44.198.160.178/SamplePage.php

Sample page

NAME	ADDRESS	
<input type="text"/>	<input type="text"/>	
<input type="button" value="Add Data"/>		
<input type="button" value="ID"/>	<input type="button" value="NAME"/>	<input type="button" value="ADDRESS"/>

Health check with id 1c5fa1ed-ba14-4f17-b55e-59b913112ee0 has been created successfully

Name	Status	Description	Alarms	ID
Samplepage	Unknown	http://abcdemployeeportal.com:80/	⚠ 1 of 1 in INSUFFICIENT...	1c5fa1ed-ba14-4f17-b55e-59b913112ee0

Click on a graph to see an expanded view. [View metrics in CloudWatch](#)

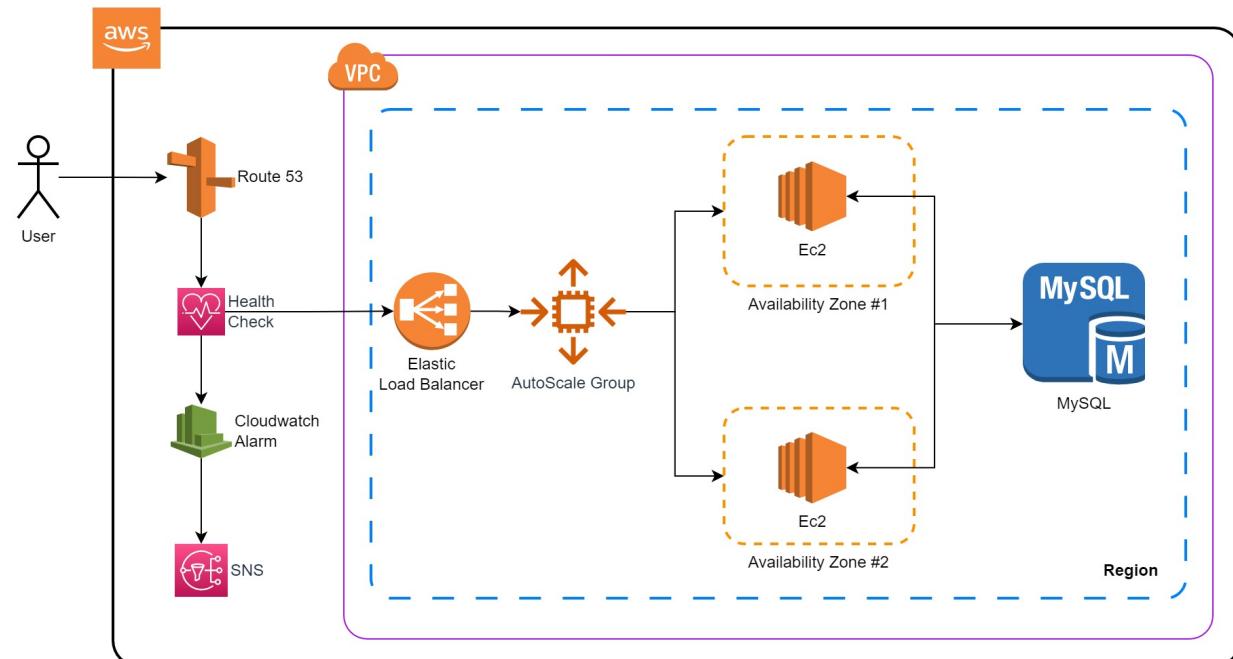
Health checks Samplepage

Time range Last hour Refresh

Health check status No data

Health checkers that report the endpoint healthy (percent) No data

Solution Architecture:



1. Architecture Overview: The architecture will consist of an EC2 instance running Apache web server, an RDS MySQL database for secure data storage, and additional services for scalability and availability.

2. EC2 Instances: Set up one or more EC2 instances to host the Apache web server. These instances will handle incoming requests from employees and serve the application. The instances should be launched in an Auto Scaling group to enable automatic scaling based on traffic demand.
3. Auto Scaling: Configure the Auto Scaling group to automatically add or remove EC2 instances based on predefined scaling policies. The policies can be based on metrics such as CPU utilization, network traffic, or request count. This ensures that the application can handle traffic surges by scaling horizontally.
4. Load Balancer: Integrate an Elastic Load Balancer (ELB) with the Auto Scaling group to distribute incoming traffic across the EC2 instances. The ELB acts as a single entry point for employees and improves the availability of the application by distributing requests evenly.
5. RDS MySQL Database: Create an RDS instance running MySQL to securely store employee inputs. Ensure that the database is provisioned with appropriate compute and storage resources based on the expected workload.
6. Security: Implement security measures such as securing the EC2 instances with appropriate firewall rules, using SSL certificates for secure communication, and applying necessary access controls to the RDS instance. Follow AWS security best practices to protect the application and data.
7. Monitoring and Logging: Configure CloudWatch to monitor key metrics of the EC2 instances, RDS database, and load balancer. Set up alarms to notify you in case of any issues or breaches. Enable logging to capture relevant logs for troubleshooting and auditing purposes.
8. Backup and Disaster Recovery: Implement regular automated backups for the RDS database to ensure data durability. Consider configuring a Multi-AZ deployment for RDS to provide high availability and automatic failover in case of a failure in the primary database.
9. Scaling Considerations: Analyze the application's performance and determine the optimal scaling thresholds based on expected traffic patterns. Regularly review and adjust the scaling policies to ensure efficient resource utilization and cost optimization.
10. Route 53: Integrate Route 53, the DNS service provided by AWS, into the architecture. Route 53 will allow you to route incoming traffic to the application's Elastic Load Balancer (ELB) and provide health checks for improved availability.
11. DNS Configuration: Create a Route 53 hosted zone for your domain and configure the necessary DNS records. Set up an "A" record or a "CNAME" record to point to the ELB's DNS name. This ensures that when employees access the application using the domain, their requests are directed to the ELB.
12. Health Checks: Configure Route 53 health checks to monitor the health of the ELB and the EC2 instances. Define health check settings to periodically send requests to the ELB and evaluate its responses. If the health checks fail for an instance or the ELB, Route 53 can automatically stop routing traffic to the unhealthy resource.
13. Routing Policies: Define routing policies in Route 53 to control the distribution of traffic among healthy resources. There are several routing policies available, such as simple, weighted, latency-based, geolocation-based, and failover. Choose the appropriate routing policy based on your requirements.
14. Simple Routing Policy: In the context of this solution, you can use the "Simple" routing policy in Route 53. With the Simple routing policy, Route 53 responds to DNS queries with all the IP addresses

associated with the ELB. The client's DNS resolver chooses one of the IP addresses randomly. This provides a basic load balancing capability across the available EC2 instances.

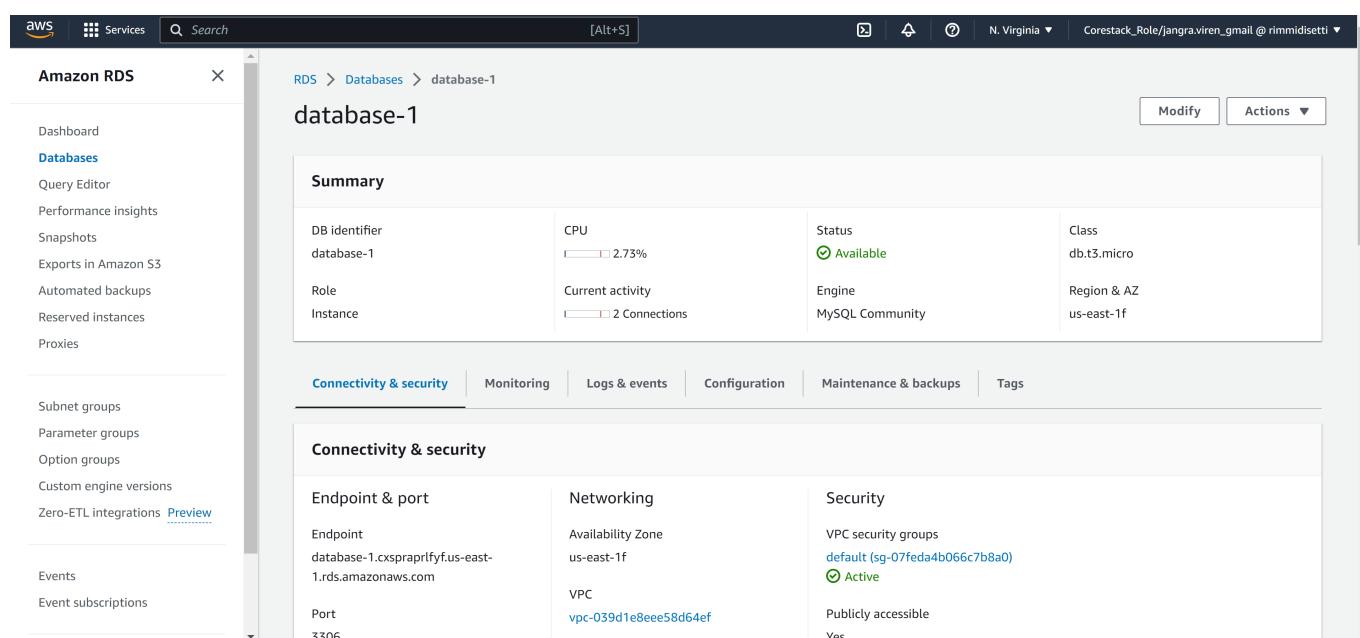
15. Health-Based Routing: Additionally, you can combine the Simple routing policy with health checks. Route 53 will only include IP addresses of the ELB associated with healthy EC2 instances in the DNS response. This further enhances the availability and reliability of the application by ensuring that traffic is only directed to healthy instances.
16. Deployment Automation: Utilize AWS CloudFormation or other deployment automation tools to define the infrastructure as code, enabling easy replication and updates of the application stack.

Possible ways of achieving the same:

1. Using Web Console and manually configure resources
2. Write CloudFormation templates, use AWS CLI and deploy through cloudformation templates.

Project Implementation:

1. RDS MySQL Instance is created opening the port 3306. In real scenarios we can have backups, snapshots and additional users for RDS access. Direct opening of the 3306 port from public won't be a good option and we should go for a Jump Server which would be inside the same VPC/Network.



The screenshot shows the AWS RDS console for a MySQL database named 'database-1'. The 'Summary' tab is selected, displaying the following information:

DB identifier	CPU	Status	Class
database-1	2.73%	Available	db.t3.micro

The 'Role' section shows 'Instance' and 'Current activity' with 2 connections. The 'Connectivity & security' tab is selected, showing the following details:

Endpoint & port	Networking	Security
Endpoint: database-1.cxspraprlyf.us-east-1.rds.amazonaws.com Port: 3306	Availability Zone: us-east-1f VPC: vpc-039d1e8eee58d64ef	VPC security groups: default (sg-07fed4b066c7b8a0) Active: Yes

2. Create a project schema/database and the required table with required columns.

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'File', 'Edit', 'View', 'Query', 'Database', 'Server', 'Tools', 'Scripting', and 'Help' are visible. The 'Database' tab is selected. On the left, the 'Navigator' pane shows 'SCHEMAS' with 'project01' expanded, containing 'Tables', 'Views', 'Stored Procedures', and 'Functions'. Under 'Tables', 'employees' is selected. The main workspace displays the 'employees' table definition. The 'Table Name' is 'employees', 'Schema' is 'project01', 'Charset/Collation' is 'utf8mb4', and 'Engine' is 'InnoDB'. The table structure includes columns: 'id' (PK, INT), 'name' (VARCHAR(45)), 'address' (VARCHAR(200)), and 'createdOn' (TIMESTAMP). Below the table definition, there are fields for 'Column Name', 'Datatype', 'PK', 'NN', 'UQ', 'B', 'UN', 'ZF', 'AI', 'G', and 'Default/Expression'. A 'Comments:' field is also present. On the right, there are sections for 'Data Type', 'Default', and 'Storage' (Virtual, Stored, Primary Key, Not Null, Unique, Binary, Unsigned, Zero Fill, Auto Increment, Generated). At the bottom, tabs for 'Columns', 'Indexes', 'Foreign Keys', 'Triggers', 'Partitioning', and 'Options' are shown, along with 'Apply' and 'Revert' buttons. The message log at the bottom right shows 'CREATE SCHEMA project01' was successful with 1 row(s) affected in 0.234 sec.

3. Hosted Zone Created: Actual hosted zone can be public and would be actual domain based. Also health check points are created for the DNS which can check the

The screenshot shows the AWS Route 53 console. The left sidebar includes 'Route 53', 'Dashboard', 'Hosted zones' (selected), 'Health checks', 'IP-based routing', 'CIDR collections', 'Traffic flow', 'Traffic policies', 'Policy records', 'Domains', 'Registered domains', 'Pending requests', 'Resolver', 'VPCs', 'Inbound endpoints', 'Outbound endpoints', 'Rules', and 'Query logging'. The main area shows a green success message: 'splproject01.com was successfully created. Now you can create records in the hosted zone to specify how you want Route 53 to route traffic for your domain.' Below this, the 'Hosted zone details' section is shown. Under 'Records (2)', there are two NS records listed:

Record name	Type	Routing policy	Alias	Value/Route traffic to	TTL (s...)
splproject01.com	NS	Simple	-	ns-1536.awsdns-00.co.uk. ns-0.awsdns-00.com. ns-1024.awsdns-00.org. ns-512.awsdns-00.net.	172800
splproject01.com	SOA	Simple	-	ns-1536.awsdns-00.co.uk. a...	900

4. Ec2 launch template created with user data script for the required files. Script includes installing apache, php and required package to connect php application to mysql database. Apart from that application files like index.html, save_data.php, get_data.php and a config.php is included in the script. Although in real world scenarios and environment based scenarios, we would use AWS Secrets manager and Database proxy for the db connectivity. But just for this project, we store credentials in plain config file.

The screenshot shows the AWS EC2 Launch Templates page. On the left, there's a sidebar with navigation links for EC2 Dashboard, EC2 Global View, Events, Instances, Launch Templates (which is selected), Images, AMIs, and Elastic Block Store. The main content area shows a table titled "Launch templates (1/1) Info". The table has columns for Launch template ID, Launch template name, Default version, and Latest version. One entry is listed: "lt-Oabcbe79338347926" under "Launch template ID", "ec2-project-01-template" under "Launch template name", "1" under "Default version", and "1" under "Latest version". Below the table, a modal window titled "ec2-project-01-template (lt-Oabcbe79338347926)" displays "Launch template details" with fields for Launch template ID, Launch template name, Default version, and Owner.

5. Application load balancer is created with target group on port 80.

The screenshot shows the AWS EC2 Load Balancers page. The sidebar includes links for EC2 Dashboard, EC2 Global View, Events, Instances, Launch Templates, Images, AMIs, and Elastic Block Store. The main area shows a table titled "Load balancers (1/1)". A filter bar at the top shows "ec2-project-01-alb". The table columns are Name, DNS name, State, VPC ID, Availability Zones, Type, and Date. One row is listed: "ec2-project-01-alb" under Name, "ec2-project-01-alb-15684..." under DNS name, "Active" under State, "vpc-039d1e8eee58d64ef" under VPC ID, "3 Availability Zones" under Availability Zones, "application" under Type, and "July (UTC)" under Date. Below the table, a modal window titled "Load balancer: ec2-project-01-alb" shows the "Details" tab, which provides information about the load balancer type (Application), status (Active), VPC (vpc-039d1e8eee58d64ef), IP address type (IPv4), scheme (Internet-facing), hosted zone (Z35SXDOTRO7X7K), availability zones (subnet-0240ab33b913a1d0c), and date created (Jul 14, 2023, 14:10 (UTC+05:30)).

6. Alias record for the load balancer is created in the hosted zone. So that this domain route all the traffic to the load balancer.

The screenshot shows the AWS Route 53 service page. On the left, there's a sidebar with navigation links like Dashboard, Hosted zones, IP-based routing, Traffic flow, Domains, Resolver, and DNS Firewall. The main area has a title bar with 'Route 53' and a search bar. A modal window at the top says 'Record for splproject01.com was successfully created.' Below it, a message says 'Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. Use "View status" button to check propagation status.' There's a 'View status' button. The main content area shows a table titled 'Records (1/3) Info' with three rows. The first row has 'Name' as 'splproject01.com', 'Type' as 'A', 'Alias' as 'No', and 'Value/Route traffic to' as 'ns-1536.awsdns-00.co.uk., ns-0.awsdns-00.com., ns-1024.awsdns-00.org., ns-512.awsdns-00.net.'. The second row has 'Name' as 'splproject01.com', 'Type' as 'A', 'Alias' as 'No', and 'Value/Route traffic to' as 'ns-1536.awsdns-00.co.uk., a...'. The third row has 'Name' as 'splproject01.com', 'Type' as 'A', 'Alias' as 'Yes', and 'Value/Route traffic to' as 'dualstack.ec2-project-01-alb...'. The right side of the screen shows 'Record details' for the first record, including 'Record name' (www.splproject01.com), 'Record type' (A), 'Value' (dualstack.ec2-project-01-alb-1568425506.us-east-1.elb.amazonaws.com.), 'Alias' (Yes), 'TTL (seconds)' (1), and 'Routing policy' (Simple).

7. DNS Routing Policy created with health check. So that Route 53 will only include healthy EC2 instances in the Load Balancer in the DNS response. This further enhances the availability and reliability of the application by ensuring that traffic is only directed to healthy instances

The screenshot shows the 'Create health check' page. It starts with a 'Step 1: Configure health check' section with two steps: 'Get notified when health check fails' and 'Configure health check'. The 'Configure health check' section has a title 'Configure health check' with a help icon. It explains that Route 53 health checks let you track the health status of your resources, such as web servers or mail servers, and take action when an outage occurs. The configuration form includes:

- Name:** splproject01-health-check
- What to monitor:** Endpoint (selected)
- Monitor an endpoint:**
 - Protocol: HTTP
 - Domain name: splproject01.com
 - Port: 80
 - Path: /index.html
- URL:** http://splproject01.com:80/index.html
- Health check type:** Basic - no additional options selected (View Pricing)

At the bottom, there are buttons for '* Required', 'Cancel', and 'Next'.

The screenshot shows the second step of the 'Create health check' wizard. It's titled 'Get notified when health check fails'. A note says: 'If you want CloudWatch to send you an Amazon SNS notification, such as an email, when the status of the health check changes to unhealthy, create an alarm and specify where to send notifications.' Below this, there's a 'Create alarm' section with a radio button set to 'Yes'. A note below it states: 'CloudWatch sends you an Amazon SNS notification whenever the status of this health check is unhealthy for at least one minute. The alarm will be located in the us-east-1 region.' There's also a 'Send notification to' section with options for 'Existing SNS topic' or 'New SNS topic'. The 'Topic name' is set to 'splproject01-health-check-sns' and the 'Recipient email addresses' is set to 'jangra.viren@gmail.com'. A note at the bottom says: 'Separate multiple addresses with a comma, a semicolon, or a space'.

Step 1: Configure health check

Step 2: Get notified when health check fails

Get notified when health check fails

If you want CloudWatch to send you an Amazon SNS notification, such as an email, when the status of the health check changes to unhealthy, create an alarm and specify where to send notifications.

Create alarm Yes No [?](#)

CloudWatch sends you an Amazon SNS notification whenever the status of this health check is unhealthy for at least one minute. The alarm will be located in the **us-east-1** region.

Send notification to Existing SNS topic New SNS topic [?](#)

Topic name * **splproject01-health-check-sns** [?](#)

Recipient email addresses * **jangra.viren@gmail.com** [?](#)

Separate multiple addresses with a comma, a semicolon, or a space

* Required [Cancel](#) [Previous](#) [Create health check](#)

The screenshot shows the 'Health checks' console. On the left is a navigation sidebar with various options like Dashboard, Hosted zones, Health checks (which is selected), IP-based routing, CIDR collections, Traffic flow, Traffic policies, Policy records, Domains, Registered domains, Pending requests, Resolver, VPCs, Inbound endpoints, Outbound endpoints, Rules, Query logging, DNS Firewall, Rule groups, Domain lists, Application Recovery Controller, and Corestack Role/jangra.viren_gmail @ rimmidisetti. The main area shows a success message: 'Health check with id 201ccb27-d9f8-458c-9447-d4083d8a8bd3 has been created successfully'. Below this is a feedback survey message: 'Health checks console feedback collection. To help us improve the Health Check user experience, please take 5 minutes to complete this survey.' A table lists the health check details:

Name	Status	Description	Alarms	ID
splproject01-health-check	Unknown	http://splproject01.com:80/index.html	⚠ 1 of 1 in INSUFFICIENT_DATA	201ccb27-d9f8-458c-9447-d4083d8a8bd3

Below the table is a 'Info' tab panel with the message: 'No health check selected.'

8. AutoScaling group is created with the required configuration. Desired capacity, min, max instances and policy could be any based on the real world situation.

The screenshot shows the AWS Auto Scaling Groups page. At the top, a green banner indicates "project-01-asg, 1 Scaling policy, 1 Notification created successfully". Below this, the "Auto Scaling groups (1/1)" section is displayed. A table lists one group: "project-01-asg" with a launch template "ec2-project-01-template | Version Default". The group has a desired capacity of 3, minimum of 2, and maximum of 5. The status is shown as "-". On the right, the Amazon Resource Name (ARN) is provided: arn:aws:autoscaling:us-east-1:40034529153:autoScalingGroup:4ee0e052-31b4-49de-a291-d397435a8c6c:autoScalingGroupName/project-01-asg.

9. Required instances automatically created and deleted based on the auto scaling configuration.

The screenshot shows the AWS Instances page. The left sidebar includes options like EC2 Dashboard, EC2 Global View, Events, Instances (selected), Instances Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, AMI Catalog, and Elastic Block Store. The main content area displays a table titled "Instances (4) Info" showing four running instances. The columns include Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4. The instances are labeled "ec2-project-01" with instance IDs i-0ba0a70b5ec451924, i-03b2b90f13265996c, i-0be764381b3f90359, and i-032ddb53db4d83b59, all running t2.micro instances in us-east-1 availability zones.

Instances (12) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
ec2-project-01	i-06d3d41ba8a93e5c7	Terminated	t2.micro	-	No alarms	us-east-1b	-
ec2-project-01	i-0f310ba7a689c8b70	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b	ec2-44-214-5
ec2-project-01	i-0f13d2fd87c262998	Terminated	t2.micro	-	No alarms	us-east-1b	-
ec2-project-01	i-0671fc35689395208	Terminated	t2.micro	-	No alarms	us-east-1b	-
ec2-project-01	i-0ba0a70b5ec451924	Terminated	t2.micro	-	No alarms	us-east-1b	-
ec2-project-01	i-03b2b90f13265996c	Terminated	t2.micro	-	No alarms	us-east-1a	-
ec2-project-01	i-000c538e988ba0233	Terminated	t2.micro	-	No alarms	us-east-1a	-
ec2-project-01	i-0be764381b3f90359	Running	t2.micro	2/2 checks passed	No alarms	us-east-1d	ec2-3-89-24
ec2-project-01	i-032ddb53db4d83b59	Terminated	t2.micro	-	No alarms	us-east-1c	-
ec2-project-01	i-08e91d55d2d3314ed	Terminated	t2.micro	-	No alarms	us-east-1c	-
ec2-project-01	i-083adff0447d5a2d7c	Running	t2.micro	2/2 checks passed	No alarms	us-east-1c	ec2-44-211-
ec2-project-01	i-095089cecebdd6c6ed	Running	t2.micro	-	No alarms	us-east-1c	ec2-52-91-2

Select an instance

10. Cloudwatch Alarm take relevant action whenever policy configuration/rule executed.

CloudWatch > Alarms

Alarms (2)

Name	State	Last state update	Conditions	Actions
TargetTracking-project-01-asg-AlarmLow-216bafdf-eb68-41d8-a794-e1f12cf7b6ef	In alarm	2023-07-14 11:36:19	CPUUtilization < 35 for 15 datapoints within 15 minutes	Actions enabled
splproject01-health-check-awsroute53-201ccb27-d9f8-458c-9447-d4083d8a8bd3-Low-HealthCheckStatus	In alarm	2023-07-14 06:06:48	HealthCheckStatus < 1 for 1 datapoints within 1 minute	Actions enabled

11. SNS Topic suscription email goes to the required consumer.

AWS Notification - Subscription Confirmation

AWS Notifications <no-reply@sns.amazonaws.com> to me

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:40034529153:splproject01-health-check-sns

To confirm this subscription, click or visit the link below (If this was in error no action is necessary):
[Confirm subscription](#)

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to [sns-opt-out](#)

12. Application is created as per the required logic. Name and Address taking post form is included. Ajax is used to submit the form data to the database. Employees datatable is created to list the submitted

database. Different instances showing the same data inserted.

The screenshot shows a web browser window titled "Project 01". The address bar indicates "Not secure | 44.200.232.10". The main content area has two sections: "Add Your Data:" and "Employees Data:". The "Add Your Data:" section contains fields for "Name" and "Address", and a blue "Add Data" button. Below it is a horizontal line. The "Employees Data:" section has a table with columns: ID, Name, Address, and CreatedOn. A message "NO RECORDS FOUND YET." is displayed in the table body.

This screenshot shows the same browser window after data has been added. A modal dialog box is open, displaying the message "44.200.232.10 says Data added successfully." with an "OK" button. The "Add Your Data" form now shows "Elon Musk" in the "Name" field and "NEW YORK" in the "Address" field. The "Employees Data" table now contains one record: ID 1, Name Virender Jangra, Address Gurgaon, and CreatedOn 2023-07-14 10:29:04.

The screenshot shows a web browser window titled "Project 01". The address bar indicates the site is not secure (44.200.232.10). The main content area contains two sections: "Add Your Data:" and "Employees Data:". The "Add Your Data:" section has input fields for Name and Address, and a blue "Add Data" button. The "Employees Data:" section displays a table with three rows of employee information.

Add Your Data:

Name: Address: **Add Data**

Employees Data:

ID	Name	Address	CreatedOn
3	Solution Architect	Simplilearn	2023-07-14 10:30:44
2	Elon Musk	New York	2023-07-14 10:29:45
1	Virender Jangra	Gurgaon	2023-07-14 10:29:04