Design Documentation for: Shirley Rodriguez
Example Social Research Organization
by Benjamin Yap

1. Executive Summary

Example Social Research Organization required a highly available, robust, scalable and secure web application hosted in AWS with hybrid connectivity to an on-premises environment.

The proposed solution uses an Application Load Balancer, Auto Scaling Group, Amazon RDS MySQL, AWS Secrets Manager and a Site-to-Site VPN connection to an on-premises Ubuntu MySQL server.

This design provides high availability, secure credential management, fault tolerance and database failover capability while meeting the customer's business and technical requirements.

On the fourth page of this design documentation (**Section 5. Network Topology Diagram/Design**), is an updated full topology diagram that shows the full Hybrid Cloud solution including VPN connection to on-premise and how database fail-over is part of the design.

2. Demo Video Demonstration

Video demonstration of the topology diagram, tour of the solution elements implemented in AWS Console, prove that the VPN Connection is up and that some kind of traffic can be successfully be sent through it, and that the web front end can fail-over successfully to access the on-premises database and send database queries to the website hosted on AWS VPC.

2.1 Demo Video Demonstration

1) Recorded 21 June 2026 (more comprehensive and shorter than the previous video) – Duration 1 hour
https://ara.au.panopto.com/Panopto/Pages/Viewer.aspx?id=fa5268e8-1d61-4000-aa94-b470003f78f5

2) Recorded 17th June 2026 (more comprehensive than previous video) – Duration 1:19:46
https://ara.au.panopto.com/Panopto/Pages/Viewer.aspx?id=842d163a-d5e8-4e23-9329-b46c009fb2e4

3) Recorded 15th June 2026 – Duration 1:00:09
https://ara.au.panopto.com/Panopto/Pages/Viewer.aspx?id=6cdb1bca-1a97-47b1-8c6e-b46a00877d39

The video demonstration includes:

- AWS VPN Tunnel Status (AWS Console) = Up.
- pfSense IPSec Tunnel Status = Established.
- Configuration: Route Tables, Security Groups, Bastion Host, Auto Scaling Group, Load Balancer, RDS Database, pfSense Firewall Rules.
- MySQL connectivity tests from AWS EC2 to the on-premises database.
- Secrets Manager manual failover demonstration.
- Example Social website successfully querying both AWS RDS and the on-premises MySQL database.
- MySQL Database name: countries.
- Created account on on-premise MySQL server:
    - Username: appuser, password: Password123!
- PHP application files: lifeexpectancy.php file and Life Expectancy query was not displaying any output after clicking Submit as the following line of code was used or not commented out – which is most likely only for authentication purposes ($_SESSION) and the page was not connected to the MySQL database:
  `$conn = new mysqli($_SESSION['ep'], $_SESSION['un'], $_SESSION['pw'], $_SESSION['db']);`
  The website is publicly accessible without authentication so does not require the use of the **`$_SESSION`** variable used to store and persist user data across multiple pages during a single website visit.

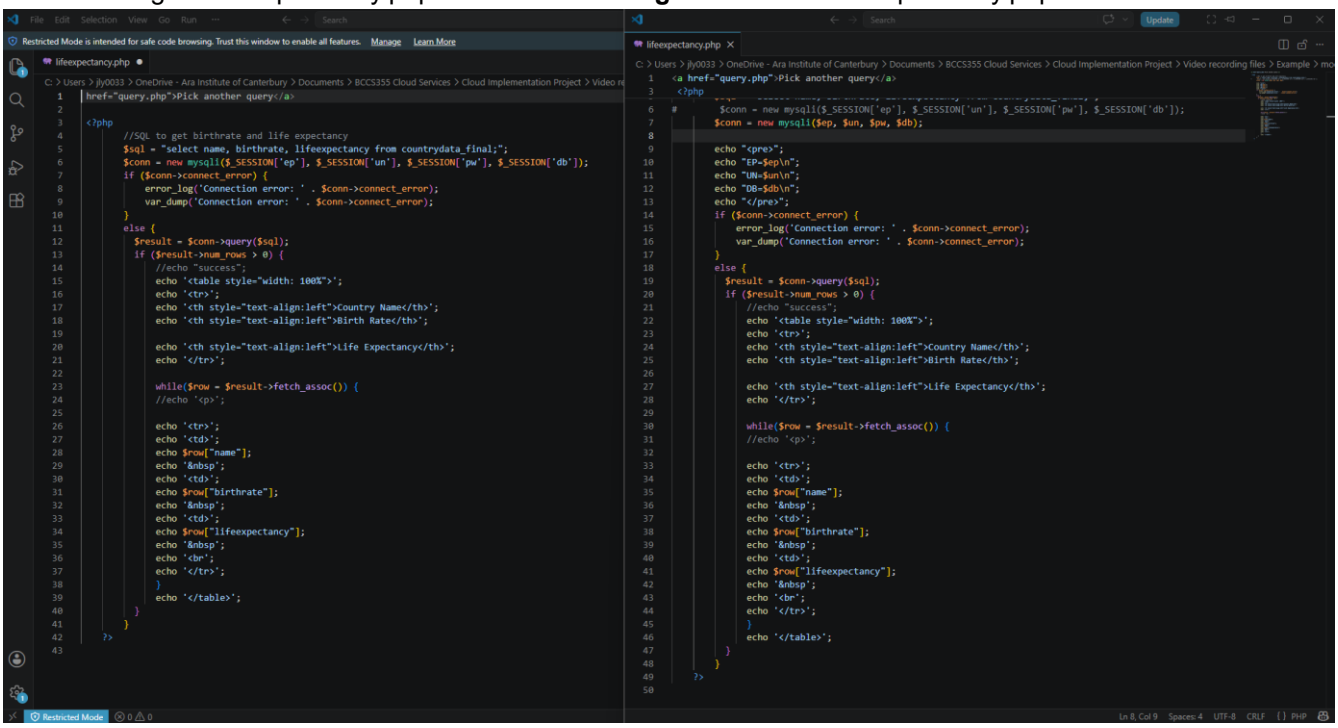
Copied this line from the other query PHP files to connect to the MySQL database to retrieve data:

```
$conn = new mysqli($ep, $un, $pw, $db);
```

Before and after:

Left: Original lifeexpectancy.php file

Right: Corrected lifeexpectancy.php file



The following code added to **gdp.php**, **lifeexpectancy.php**, **mobile.php**, **mortality.php**, **population.php** files below the new database connection command/line to display the database host (`$ep`), database username (`$un`) and the database name of `countries` (`$db`).

```
echo "<pre>";
echo "EP=$ep\n";
echo "UN=$un\n";
echo "DB=$db\n";
echo "</pre>";
```

Purpose: To verify which database the queries are being retrieved from and also to verify that database fail-over is working as expected.

3. Customer Requirements

- Highly available website application through a load balancer & provide anonymous access to web users
- Database hosted within AWS – secure access to the database
- Secure storage of database credentials using the Secrets Manager
- Ability to scale web servers automatically
- Secure remote administration
- Hybrid Cloud connectivity
- On-premises database failover capability
- Demonstration of database access through a VPN connection

4. Non-Functional Requirements

Requirement	Target
Availability	99.9%
Scalability	Up to 4 web servers
Security	Secrets Manager and Security Groups
Recovery	Manual failover to on-premises database from AWS RDS database
Performance	Load balanced across multiple EC2 instances

5. Network Topology Diagram/Design

Figure 1 – Starting Architecture Topology

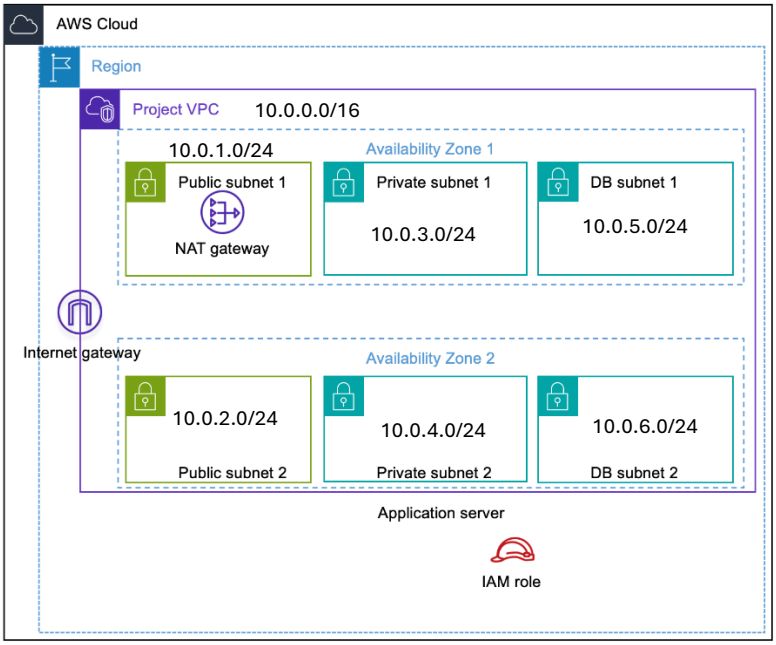
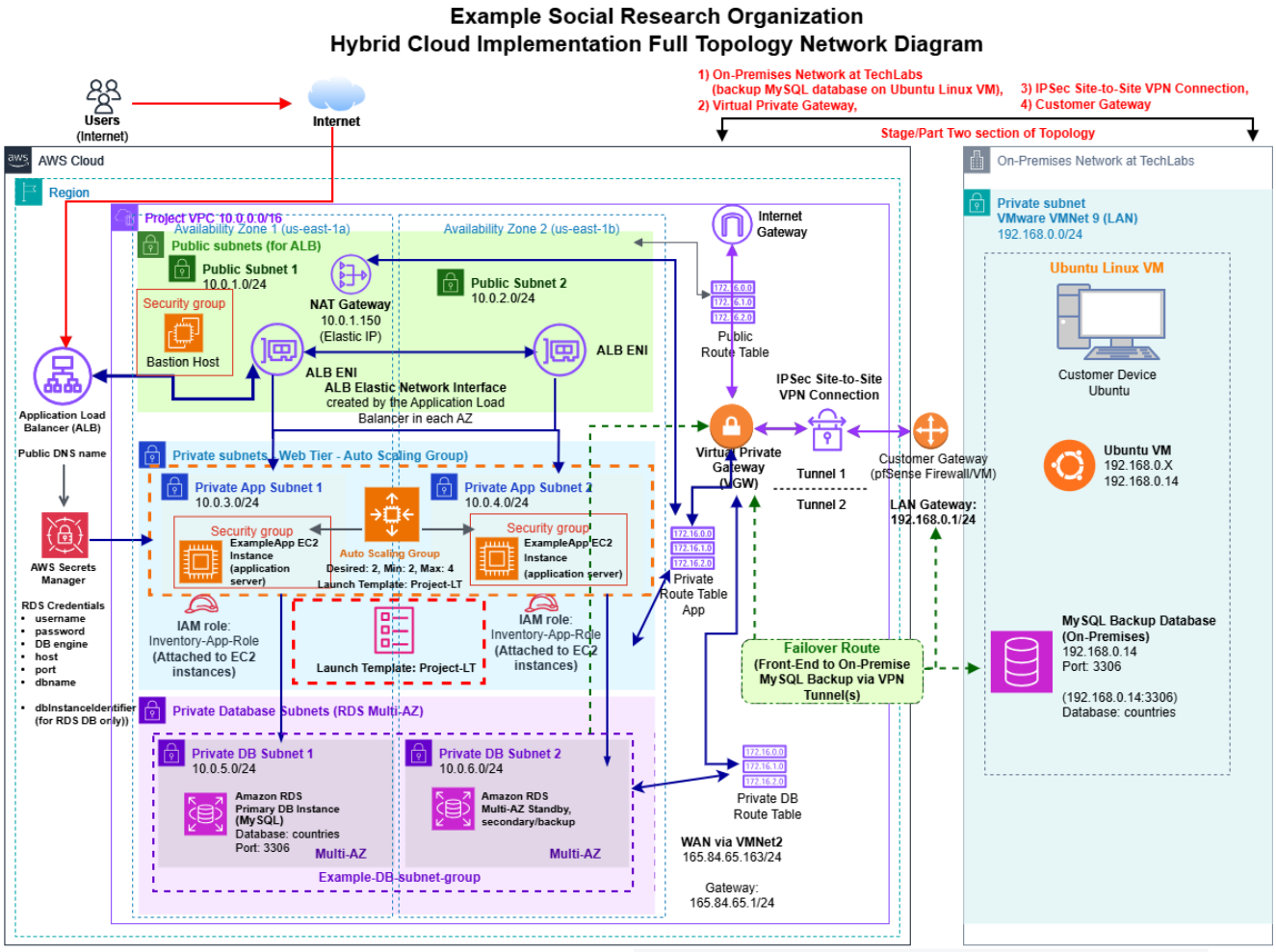
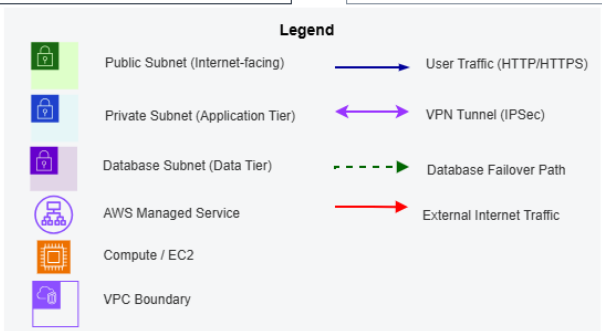
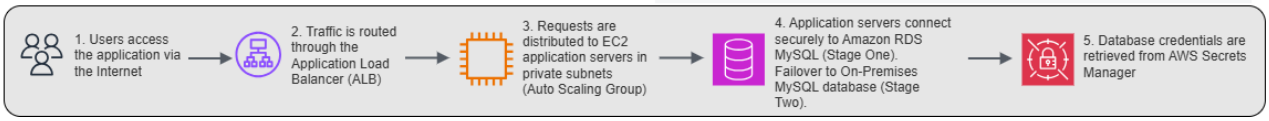


Figure 2 – Full Example Social Hybrid Cloud Topology (with changes made. Updated 20 June 2026)



AWS VPC - CIDR & Subnets			
Component	**Description**	**CIDR Block**	**AZ**
VPC	Example Social VPC	10.0.0.0/16	-
Public Subnet 1	ALB	10.0.1.0/24	us-east-1a
Public Subnet 2	ALB	10.0.2.0/24	us-east-1b
Private App Subnet 1	Web Tier	10.0.3.0/24	us-east-1a
Private App Subnet 2	Web Tier	10.0.4.0/24	us-east-1b
Private DB Subnet 1	RDS Primary	10.0.5.0/24	us-east-1a
Private DB Subnet 2	RDS Standby	10.0.6.0/24	us-east-1b





5.1 Routing Tables

Public Route Table	
VPC for this route table: Project VPC	
Routes	
Destination	**Target**
0.0.0.0/0	Internet gateway igw-0278040a5fd37c27a
Subnet associations	
Public Subnet 1	Public Subnet 2
Route propagation	
VPG-example-social	

Private Route Table App		
VPC for this route table: Project VPC		
Routes		
Destination	**Target**	**Purpose**
0.0.0.0/0	NAT gateway: Lab-NATGW nat-04ea5826ef4259fb6	Internet Access
10.0.0.0/16	Local	Local VPC
192.168.0.0/24	Virtual Private Gateway: VPG-example-social vgw-0cbc6238ea23c521d	On-Premises via VPN
Subnet associations		
Private Subnet 1	Private Subnet 2	
Route propagation		
VPG-example-social		

Private DB Route Table		
VPC for this route table: Project VPC		
Routes		
Destination	**Target**	**Purpose**
0.0.0.0/16	Local	Local VPC
192.168.0.0/24	Virtual Private Gateway: VPG-example-social vgw-0cbc6238ea23c521d	On-Premises via VPN
Subnet associations		
Private DB Subnet 1	Private DB Subnet 2	
Route propagation		
VPG-example-social		

5.2 Security Groups

Inventory-App: Enable access to App			
Inbound Rules:			
Type	**Protocol**	**Port Range**	**Source**
SSH	TCP	22	Bastion Host
HTTP	TCP	80	ALBSG
HTTPS	TCP	443	ALBSG
Outbound Rules:			
All traffic	All	All	0.0.0.0/0

ExampleDB-SG: Enable access to MySQL			
Inbound Rules:			
Type	**Protocol**	**Port Range**	**Source**
MySQL/Aurora	TCP	3306	192.168.0.0/24
MySQL/Aurora	TCP	3306	Inventory-App
Outbound Rules:			
All traffic	All traffic	All traffic	All traffic (0.0.0.0/0)

ALB-SG: Port 80: To access website via Application Load Balancer			
Inbound Rules:			
Type	**Protocol**	**Port Range**	**Source**
HTTP	TCP	80	0.0.0.0/0
HTTPS	TCP	443	0.0.0.0/0
Outbound Rules:			
All traffic	All traffic	All traffic	All traffic (0.0.0.0/0)

5.3 Design Overview: AWS Services Used

Component	Purpose
VPC	Network isolation
Internet Gateway	Internet access
NAT Gateway	Outbound internet access for private instances
Application Load Balancer/ALB	Load balances traffic. Traffic distribution
Auto Scaling Group	Automatically scale and replace web servers
EC2	Web application hosting
RDS MySQL	Managed and primary database on AWS
Secrets Manager	Secure credential storage
IAM Roles	Secure AWS service access
Site-to-Site VPN	Hybrid connectivity between AWS Cloud and On-Premises
Customer Gateway	On-premises VPN endpoint implemented using pfSense.
Virtual Private Gateway	AWS-side VPN endpoint attached to the VPC.
On-Premises MySQL	Database failover target located and installed on Ubuntu VM on-premises for backup of database storage.
CloudWatch	Monitoring and logging

5.4 High Availability Features

Feature	Benefit
Multi-AZ Deployment	Survives Availability Zone failure
Application Load Balancer	Distributes traffic
Auto Scaling	Replaces failed EC2 instances
Multi-AZ RDS	Database redundancy
Dual VPN Tunnels	VPN redundancy

6. Design Decisions

6.1 Multi-Availability Zone Design

Decision: Two Availability Zones were used

Reason: A single Availability Zone failure would cause service disruption.

Benefit: Provides high availability and fault tolerance.

6.2 Application Load Balancer (ALB)

Decision: An Application Load Balancer was deployed in public subnets.

Reason: An ALB distributes traffic across multiple web servers.

Benefit: Improves availability and scalability.

6.3 Auto Scaling Group

Decision: The Web Tier uses an Auto Scaling Group – in the **Private Subnets**.

Configuration
Minimum Capacity: 2
Desired Capacity: 2
Maximum Capacity: 4

An EC2 launch template named Example-LT is provided in the AWS account. It already has both the SQL dump file and the unzipped website files built into it.

Reason: Automatically replaces failed EC2 instances and allows scaling during increases in demand.

Benefit: Improves resilience and availability.

6.4 Private Application Subnets

Decision: Web servers were deployed into Private Subnets.

Reason: Direct Internet access to web servers increases attack surface – high risk to external attacks by “bad actors” from external networks.

Benefit: Improved security.

6.5 Bastion Host

Decision: A Bastion Host was deployed in a Public Subnet.

Reason: Provides controlled SSH access from the on-premises Ubuntu VM to private EC2 instances hosted within the AWS VPC.

Benefit: Follows AWS best practices for SSH access to EC2 instances.

6.6 Secrets Manager

Decision: Database credentials were stored in Secrets Manager – to connect to the database to execute and display queries on the front-end of the web application/website.

Reason: Avoid hardcoding credentials inside application code.

Benefit: Improves security and supports failover for Shirley’s web front-end website to access a backup database hosted on-premises.

6.7 RDS Database (MySQL)

Decision: Amazon RDS database running MySQL was selected.

Reason: Managed database service reduces administrative overhead. Also familiar with using MySQL for personal projects on Linux based servers.

Benefit: Automatic backups and improved availability.

6.8 On-Premises Virtual Machine

Decision: Chose Ubuntu Linux over Microsoft Windows 10 or 11 Virtual Machine endpoint on-premises.

Reason: Ubuntu Linux was selected because it provides a stable, secure and lightweight server operating system commonly used in enterprise and cloud environments. It requires fewer system resources than Windows and offers strong support for MySQL and networking services.

Benefit: Reduced licensing costs, lower resource consumption, improved security and easier server administration.

7. Hybrid Cloud Design

7.1 Site-to-Site VPN

Decision: An IPSec VPN connection was implemented between AWS (AWS Cloud) and Ara TechLabs (On-Premises).

Configuration

Component	Value
AWS VPC	10.0.0.0/16
VMNet 9	192.168.0.0/24
VPN Type	IPSec
Tunnels	2

Reason: Securely connects Cloud (VMNet 2 – WAN) and On-Premises resources/networks (VMNet 9 – LAN).

Benefit: Creates a Hybrid Cloud architecture as required and specified in instructions.

7.2 On-Premises Database

Decision: A MySQL server was deployed on Ubuntu

IP address	192.168.0.14

Purpose: Provides a failover database if AWS RDS becomes unavailable.

7.3 Route Table Configuration 1

<table>
<tr><td>Decision:</td><td>A route was added to the App Tier Route Table

Private Route Table App<table><tr><th>Destination</th><th>Target</th></tr><tr><td>192.168.0.0/24</td><td>Virtual Private Gateway</td></tr></table></td></tr>
<tr><td>Reason:</td><td>Allows Example Social application servers to communicate with the on-premises network (192.168.0.0/24).</td></tr>
<tr><td>Benefit:</td><td>Required for application connectivity to the on-premises MySQL database during failover operations – example when the RDS database is stopped or not functioning properly or undergoing maintenance.</td></tr>
</table>

7.4 Route Table Configuration 2

<table>
<tr><td>Decision:</td><td>A route was added to the Database Tier Route Table

Private DB Route Table<table><tr><th>Destination</th><th>Target</th></tr><tr><td>192.168.0.0/24</td><td>Virtual Private Gateway</td></tr></table></td></tr>
<tr><td>Reason:</td><td>Allows Example Social application servers to communicate with the on-premises network (192.168.0.0/24).</td></tr>
<tr><td>Benefit:</td><td>Required for application connectivity to the on-premises MySQL database during failover operations – example when the RDS database is stopped or not functioning properly or undergoing maintenance.</td></tr>
</table>

7.5 Hybrid Connectivity Validation

The Hybrid Cloud solution was validated by establishing a Site-to-Site IPSec VPN connection between AWS and the Ara TechLabs on-premises environment.

Connectivity from AWS Cloud to on-site was verified via the Front-End Example Social website performing the Queries of the on-premises MySQL database hosted/installed on Ubuntu Linux Virtual Machine (192.168.0.14)

Successful database connections confirmed that traffic was traversing the VPN tunnel(s) and that routing, Firewall Rules and Security Group configurations were functioning correctly.

This validation demonstrates that the AWS Application Tier can communicate with resources located within the on-premises VMNet9 network and confirms successful implementation of a Hybrid Cloud architecture.

8. Security Controls

Control	Implementation
Network Segmentation	Public, Private and DB Subnets
Security Groups	Least privilege access
Secrets Manager	Secure credential storage
IAM Roles	No hardcoded AWS credentials
Private Subnets	Web servers not internet accessible
VPN Encryption	IPSec encrypted tunnels
Bastion Host	Controlled administration access

9. Testing and Validation

Testing Performed

Test	Expected Result	Result
ALB Website Access	Website accessible via ALB DNS	Passed
SSH Bastion Access	SSH access to private EC2	Passed
RDS Database Connectivity	Web application successfully connects to the Amazon RDS MySQL database and retrieves query results.	Passed
Secrets Manager Retrieval	Application successfully retrieves database host, username, password and database name from AWS Secrets Manager.	Passed
VPN Tunnel Connectivity	Tunnel status UP	Passed
On-Premises Database Connectivity	Web application (PHP files) successfully connects to the MySQL database hosted on the Ubuntu on-premises server through the Site-to-Site VPN connection and retrieves query results.	Passed
Database Failover	After updating AWS Secrets Manager to use the on-premises MySQL endpoint (`192.168.0.14`) from the RDS database endpoint (`example-social-rds.c53ujk05hh3k.us-east-1.rds.amazonaws.com`), the Example Social website successfully retrieves query results through the VPN connection.	Passed
Auto Scaling Group Health	Auto Scaling Group maintains two **Healthy** EC2 instances across private subnets and automatically launches replacement instances if an instance becomes **Unhealthy** or is terminated to maintain the `Desired Capacity` of 2.	Passed

10. Risks and Mitigations

Risks	Mitigation
RDS Failure	On-premises MySQL failover
EC2 Failure	Auto Scaling replacement
AZ Failure	Multi-AZ architecture
Credential Exposure	Secrets Manager
VPN Tunnel Failure	Dual IPSec tunnels

11. Recovery Objectives

Metric	Value
RTO (Recovery Time Objective)	Under 15 minutes
RPO (Recovery Point Objective)	Depends on latest database backup

12. Database Failover Design

Normal Operation: User → ALB → Web Tier → Secrets Manager (host pointing to the RDS Endpoint on AWS) → RDS MySQL

Failover Operation: User → ALB → Web Tier → Secrets Manager → VPN → 192.168.0.14 MySQL
(in the Secrets Manager: host pointing to the On-Premises IP address of the Ubuntu VM)

Design Decision: Update the “host” value in AWS Secrets Manager to point to 192.168.0.14 (or 192.168.0.X – with X being the latest IPv4 address of the Ubuntu VM host on-premises)

Instead of example-social-rds.c53ujk05hh3k.us-east-1.rds.amazonaws.com (original host value pointing to the AWS RDS database).

Failover Procedure

1. Detect RDS outage.
2. Verify VPN connectivity.
3. Update AWS Secrets Manager host value, database username and password.
4. Point application from RDS MySQL database to on-premises MySQL server (host/IP address).
5. Verify front-end database connectivity.
6. Restore RDS service when available.
7. Repoint Secrets Manager to RDS endpoint.

13. Cost

Cost optimisation was considered by using **t2.micro** `ExampleAPP` EC2 instances in both Private Subnet 1 and Private Subnet 2, and limiting Auto Scaling capacity to four instances unless demand increases.

t3.micro instance class for the RDS database named `example-social-rds`.

14. Assumptions

- **Application is deployed in one AWS Region:** us-east-1 (North Virginia).
- VPN tunnel(s) remain operational.
- The website does not need to be available over HTTPS or a custom domain name.
- The solution is deployed on Amazon Linux 2023 machines by using the provided PHP code. A template (Project-LT) already provided to launch EC2 instances.
- Solution uses all the three pairs of subnets provisioned.
- Solution uses the available AWS Identity and Access Management (IAM) role.
- The database is hosted in multiple Availability Zones for redundancy and scalability.
- Database connection information stored in the AWS Secrets Manager.
- The website is publicly accessible without authentication.
- On-Premises Ubuntu server remains powered on.
- Internet connectivity exists between AWS and On-Premises environments.
- Database backups are performed regularly.

15. Future Improvements

- Implement automatic database failover using AWS Lambda and Secrets Manager without requiring manually updating the Secrets Manager.
- Enable CloudWatch alarms and SNS notifications.
- Implement AWS WAF for web application protection.
- Enable RDS automated snapshots.
- Implement database replication between RDS and on-premises MySQL.
- Use AWS Systems Manager Session Manager to reduce reliance on Bastion Host.

16. Conclusion

Part 1:

- Provide secure hosting of the MySQL database.
- Provide secure access to the database.
- Provide anonymous access to web users.
- Run the website on a t2.micro EC2 instance running in private subnets and provide Secure Shell (SSH) access to administrators.
- Provide high availability to the website through a load balancer.
- Provide automatic scaling that uses a launch template.

Part 2 Extended with the following solutions:

- A highly available AWS environment was deployed to work with or connect to an on-premises environment.
- Auto Scaling and Load Balancing were implemented.
- Database credentials were secured using Secrets Manager.
- A Hybrid Cloud architecture was successfully implemented.
- Site-to-Site VPN connectivity was established.
- Example Social application can failover to an on-premises MySQL database hosted on VMNet 9 LAN.