

A Project Documentation on
“Highly Available Web Application with
Secure VPC, S3 Storage & SNS Alerts”

Project Objective

Design and deploy a secure, scalable web application on AWS using:

- EC2 for compute
- S3 for static content & backups
- VPC for networking & security
- SNS for system notifications and alerts

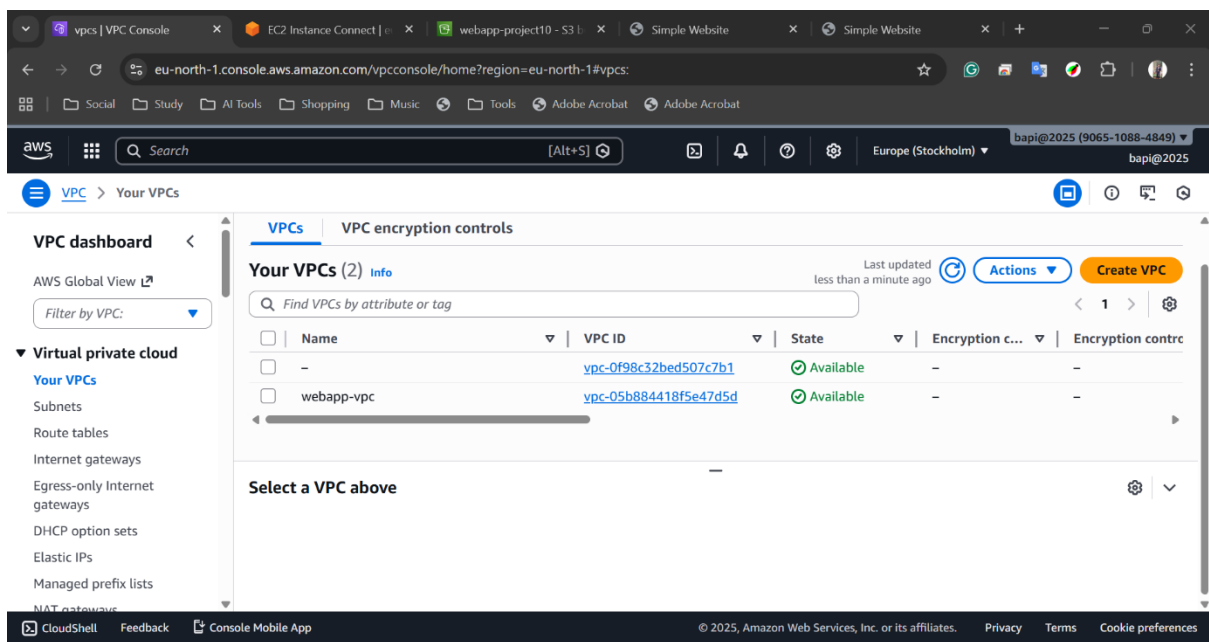
Steps:-

Step 1: Create a Custom VPC

- Go to VPC → Create VPC
- Select VPC only

Enter:

- Name: **webapp-vpc**
- IPv4 CIDR: 10.0.0.0/16
- Click Create VPC



Step 2: Create a Public Subnet

- Go to VPC → Subnets → Create subnet
- Select webapp-vpc

Enter:

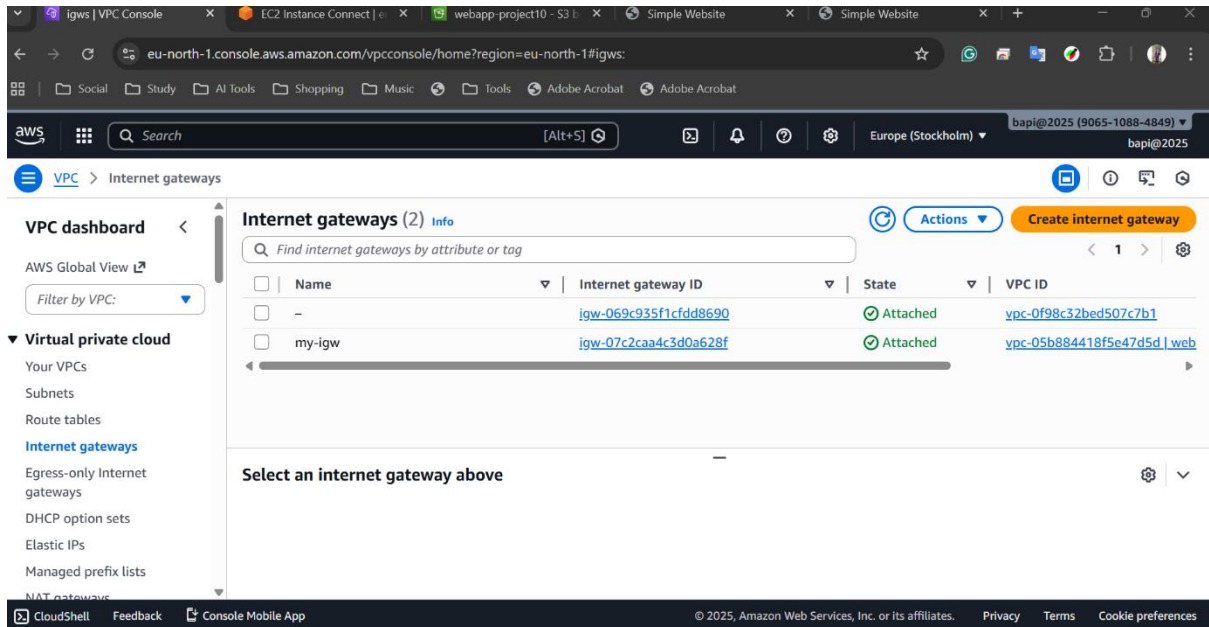
- Subnet name: **public-subnet**
- Availability Zone: any (e.g., us-east-1a)
- CIDR: 10.0.1.0/24
- Enable Auto-assign public IPv4
- Click Create subnet

The screenshot shows the AWS VPC console interface. The left sidebar contains the 'VPC dashboard' and a list of VPC resources under 'Virtual private cloud'. The main content area is titled 'Subnets (4)' and includes a search bar and a table of subnets. The table has columns for Name, Subnet ID, State, and VPC. The first row shows a 'public subnet' with Subnet ID 'subnet-04034bca7875885a6', State 'Available', and VPC 'vpc-05b884418f5e47d5d'. The other three subnets are unnamed and also in an 'Available' state, associated with VPC 'vpc-0f98c32bed507c7b1'. A 'Create subnet' button is visible in the top right corner.

Name	Subnet ID	State	VPC
public subnet	subnet-04034bca7875885a6	Available	vpc-05b884418f5e47d5d
-	subnet-0bb0dcd11610113d	Available	vpc-0f98c32bed507c7b1
-	subnet-09859e0fffbef80b	Available	vpc-0f98c32bed507c7b1
-	subnet-09dba59bcfe32edb8	Available	vpc-0f98c32bed507c7b1

Step 3: Create and Attach Internet Gateway

- Go to VPC → Internet Gateways
- Click Create internet gateway
- Name: **my-igw**
- Attach it to webapp-vpc



Step 4: Configure Route Table

- Go to VPC → Route Tables

Create a route table:

- Name: **route1**
- VPC: webapp-vpc

Add route:

- Destination: 0.0.0.0/0
- Target: Internet Gateway
- Associate the route table with Public-Subnet

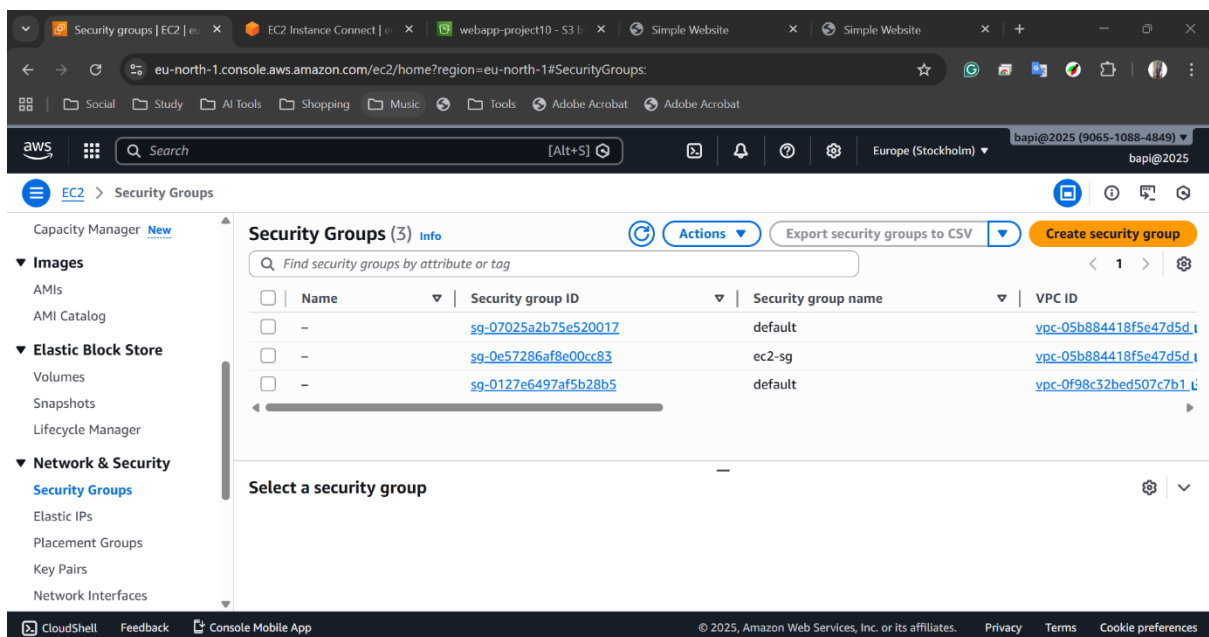
The screenshot shows the AWS Management Console interface for the 'Route tables' section. The left sidebar contains the 'VPC dashboard' and 'Virtual private cloud' navigation menu. The main content area displays a list of route tables under the heading 'Route tables (1/3) Info'. A table lists three route tables, with 'route1' (ID: rtb-044b06d436ec88905) selected. Below the list, the details for 'route1' are shown, including a table of routes. The routes table has columns for Destination, Target, Status, Propagated, and Route Origin. Two routes are listed: 0.0.0.0/0 pointing to an Internet Gateway (igw-07c2caa4c3d0a6...) and 10.0.0.0/16 pointing to 'local'. Both routes are 'Active'.

Name	Route table ID	Explicit subnet associ...	Edge associations	Main
-	rtb-0a24d9e0f6f4f271e	-	-	Yes
-	rtb-0a36c70e52b7a6d06	-	-	Yes
route1	rtb-044b06d436ec88905	subnet-04034bca787588...	-	No

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-07c2caa4c3d0a6...	Active	No	Create Route
10.0.0.0/16	local	Active	No	Create Route Table

Step 5: Create Security Group

- Go to EC2 → Security Groups → Create
- Name: **ec2_sg**
- VPC: webapp-vpc
- Inbound rules:
- HTTP – Port 80 – Source: 0.0.0.0/0
- SSH – Port 22 – Source: My IP
- Outbound: Allow all traffic
- Create security group



Step 6: Launch EC2 Instance

- Go to EC2 → Launch Instance
- Name: **webapp**
- AMI: Amazon Linux 2
- Instance type: t3.micro (Free tier)
- Key pair: created a project.pem
- Network settings:
 - VPC: webapp-vpc
 - Subnet: public-subnet
 - Auto-assign Public IP: Enabled
- Security group: ec2_sg
- User data:

```
#!/bin/bash
```

```
yum update -y
```

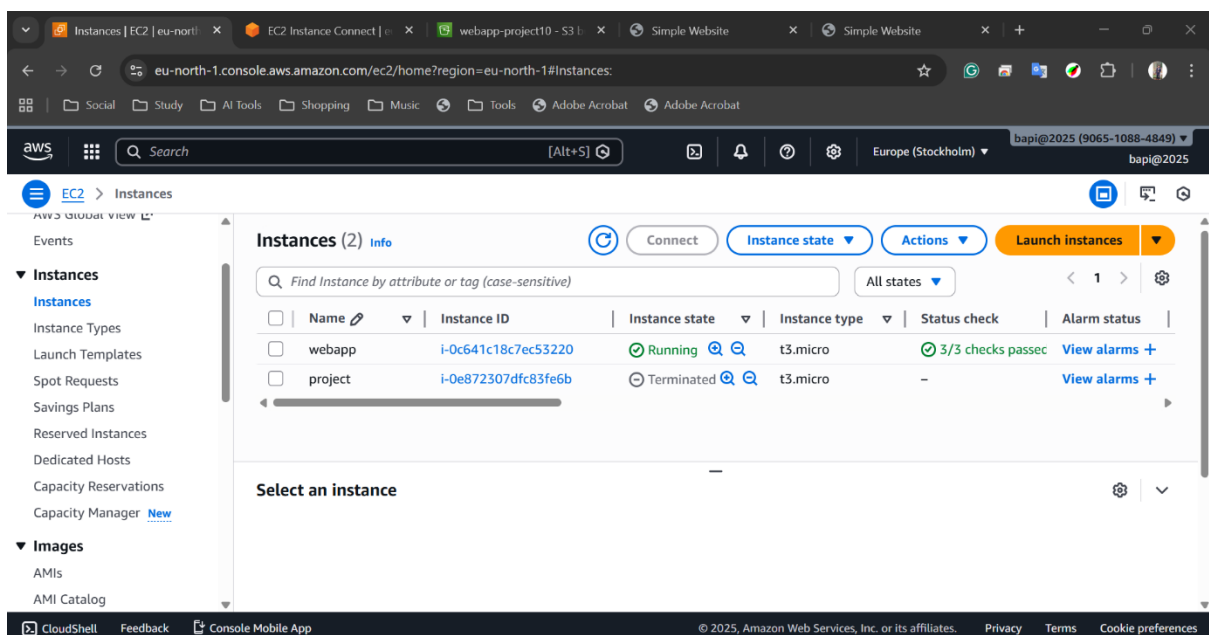
```
yum install -y httpd
```

```
systemctl start httpd
```

```
systemctl enable httpd
```

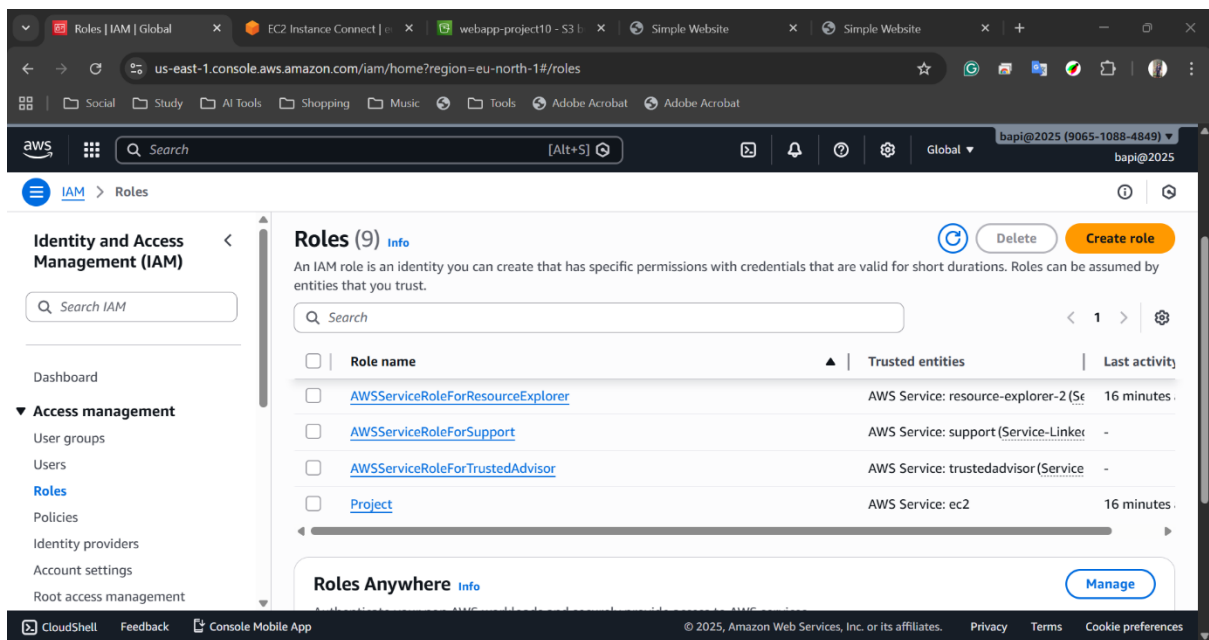
```
echo "Welcome to Highly Available Web App" > /var/www/html/index.html
```

- Launch instance



Step 7: Create IAM Role

- Go to IAM → Roles
- Create Role
- Trusted Entity: AWS Service
- Use case: EC2
- Permission policies: AmazonS3FullAccess
- Name: project
- Create Role

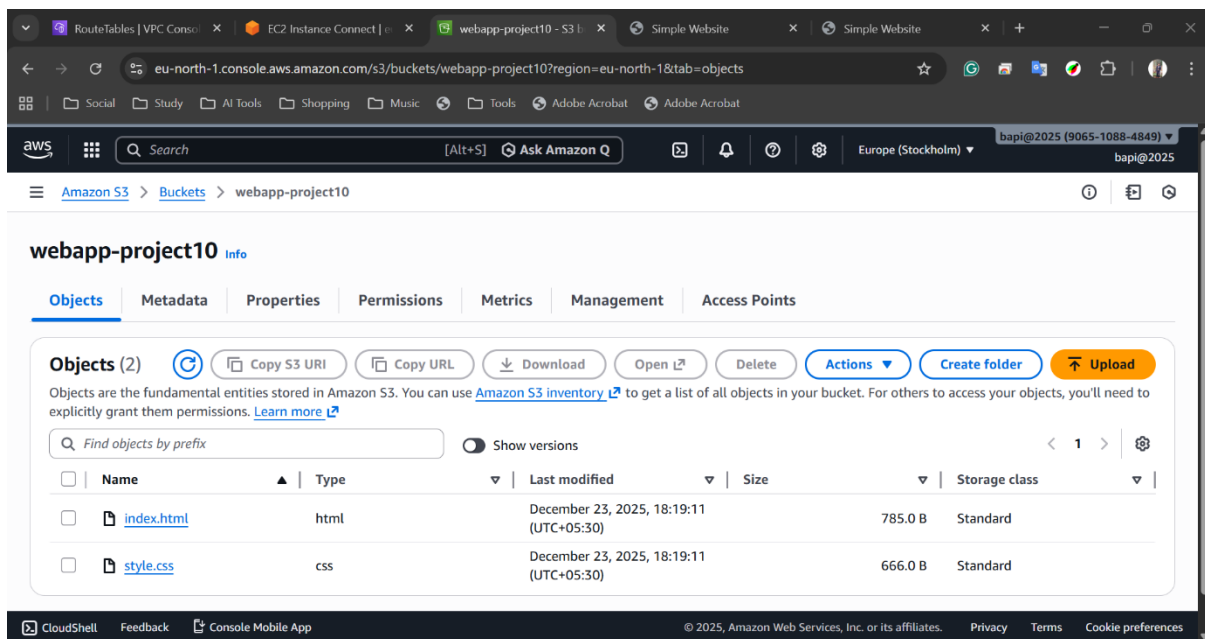


Step 8: Create S3 Bucket

- Go to S3 → Create bucket
- Bucket name: **webapp-project10**
- Region: same as EC2
- Enable:
- Versioning
- Block public access
- Create bucket

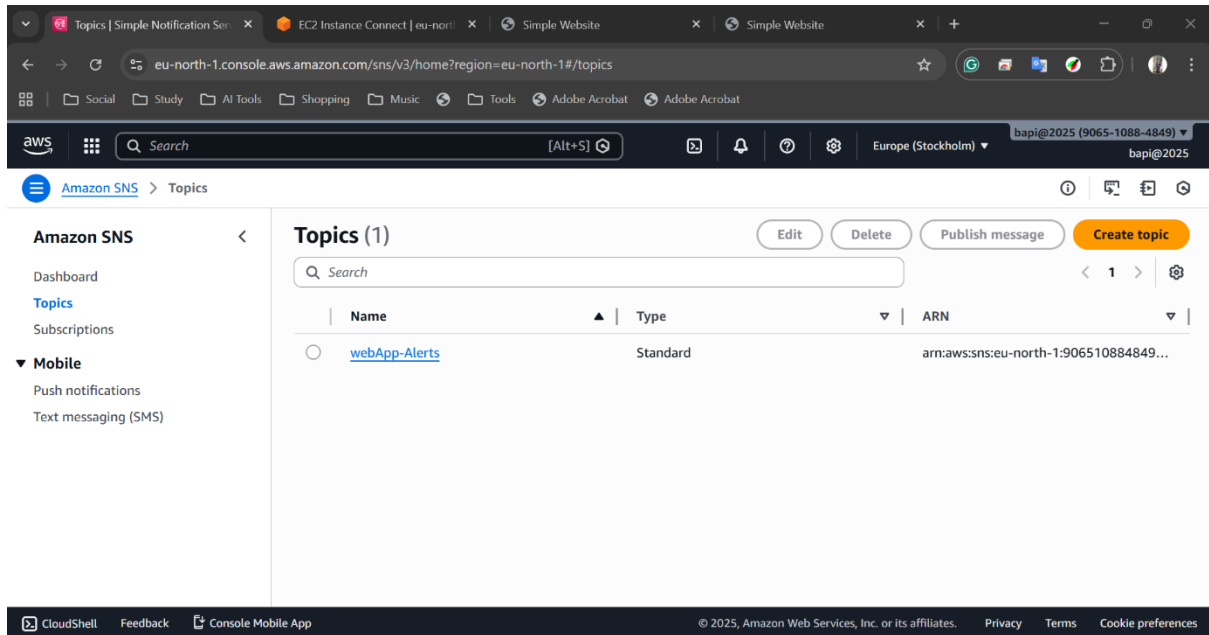
Step 9: Upload Static Content to S3

- Open the S3 bucket
- Upload:
 - index.html
 - style.css
 - script.js



Step 10: Create SNS Topic

- Go to SNS → Topics
- Click Create topic
- Type: Standard
- Name: **webapp-alerts**
- Create topic



Step 11: Subscribe to SNS Topic

- Open webapp-alerts
- Click Create subscription
- Protocol: Email
- Endpoint: your email address → dishahota23@gmail.com
- Confirm subscription from email

The screenshot shows the AWS Management Console for the SNS Subscriptions page. The left sidebar contains navigation links for Dashboard, Topics, Subscriptions, Mobile, Push notifications, and Text messaging (SMS). The main content area shows a table with one subscription. The table has columns for ID, Endpoint, Status, Protocol, and Topic. The subscription is confirmed and uses the EMAIL protocol for the webApp-Alerts topic.

ID	Endpoint	Status	Protocol	Topic
cd57132d-3657-44e0-a2b6-cf94f9ec091f	anandrajakeshri45...	Confirmed	EMAIL	webApp-Alerts

The screenshot shows an email confirmation from AWS Simple Notification Service. The email confirms the subscription and provides the subscription ID: arn:aws:sns:eu-north-1:906510884849:webApp-Alerts:cd57132d-3657-44e0-a2b6-cf94f9ec091f. It also includes a link to unsubscribe.

Subscription confirmed!

You have successfully subscribed.

Your subscription's Id is:
arn:aws:sns:eu-north-1:906510884849:webApp-Alerts:cd57132d-3657-44e0-a2b6-cf94f9ec091f

If it was not your intention to subscribe, [click here to unsubscribe](#).

Step 12: Connect SNS with Monitoring

- Go to CloudWatch
- Create alarm (example):
- EC2 CPU utilization > 80%
- Set alarm action → SNS topic
- Save alarm

The screenshot shows the AWS CloudWatch Alarms console in the eu-north-1 region. A green notification banner at the top states "Successfully created alarm webapp-alarma." with a "View alarm" button. The left sidebar shows the "Alarms" section selected under "CloudWatch". The main area displays "Alarms (1)" with a table containing one alarm:

<input type="checkbox"/>	Name	State	Last state update (UTC)	Conditions
<input type="checkbox"/>	webapp-alarma	OK	2025-12-23 13:45:01	CPUUtilization > 10000 for 1 datapoints with 5 minutes

Additional UI elements include a search bar, filters for "Alarm state: Any" and "Alarm type: Any", and a "Create alarm" button.

Step 13: Test the Application

- Copy EC2 public IP → **16.170.223.116**
- Paste into browser → Web page loads
- Stop EC2 instance → verify SNS alert (if alarm configured)
- Check S3 uploads

The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, a search bar, and the user's profile (bapi@2025). The main content area displays the EC2 Instance Connect terminal for an instance named 'webapp-project10'. The terminal output shows the Amazon Linux 2023 logo and the URL <https://aws.amazon.com/linux/amazon-linux-2023>. Below the terminal, the instance details for 'i-0c641c18c7ec53220 (webapp)' are shown, including the public IP address '16.170.223.116' and private IP address '10.0.0.71'. The bottom of the console shows the 'CloudShell' button and the footer with copyright information.

The screenshot shows a web browser window with the address bar displaying '16.170.223.116'. The website has a blue header with the text 'Welcome to My Simple Website' and 'A clean starter template'. Below the header, there is a section titled 'About This Site' with the text 'This is a minimal updated website built using plain HTML, CSS, and JavaScript. For checking Versioning'. A blue button labeled 'Click Me' is positioned below the text.