

Project name:

Harnessing EC2, VPC & MobaXterm for Cutting-Edge Entity Hosting

Project done by:

Pavithra.R & Godwin.K

Department: Computer science engineering (CSE)

Year: III

Table of Contents

Scenario 1: Secure Internal Application Hosting for a Financial Institution

* Project Objective-----	page 03
* Project Scope-----	page 04
* Project Timeline-----	page 05
* Required Resources-----	page 06

Scenario 2: High-Availability Web Hosting for an E-commerce Platform

* Project Objective-----	page 08
* Project Scope-----	page 09
* Project Timeline-----	page 10
* Required Resources-----	page 11

Scenario 3: Custom Web Application Deployment for a Start-Up

* Project Objective-----	page 13
* Project Scope-----	page 14
* Project Timeline-----	page 14
* Required Resources-----	page 15

Harnessing EC2, VPC & MobaXterm for Cutting-Edge Entity Hosting

* Architecture-----	page 16
* Pre-requisites-----	page 16
* Project-Flow-----	page 17
* Create an AWS account and login to console-----	page 17
* Setting up AWS EC2 instance-----	page 18
* Install and Configure MobaXterm-----	page 22
* Configure Web Server On EC2-----	page 24
* Deploy EduBridge Website Code-----	page 27
* Launching The Website-----	page 28
* Monitor And Optimize-----	page 29

Scenario 1:

Secure Internal Application Hosting for a Financial Institution

A financial institution needs to host a secure internal application that handles sensitive client data. The project utilizes EC2 instances within a VPC to create isolated environments for hosting the application.

MobaXterm is employed for secure remote access and management of these instances, ensuring that only authorized personnel can deploy updates or perform maintenance. This setup provides the institution with a highly secure, scalable, and easily manageable hosting environment, compliant with financial regulations.

Project Objective:

To establish a highly secure, scalable, and compliant internal application hosting environment for a financial institution by leveraging AWS EC2 instances within a VPC, secured by MobaXterm for remote access, to ensure the confidentiality, integrity, and availability of sensitive client data.

Key Objectives:

- * Security:

- * Implement robust security measures to protect sensitive client data, including:

- * Strong access controls and authentication mechanisms.

- * Encryption of data both at rest and in transit.

- * Regular security audits and vulnerability assessments.

- * Compliance with industry standards and regulations (e.g., PCI DSS, HIPAA).

- * Scalability:

- * Design the infrastructure to be scalable to accommodate future growth and increased workload.

- * Implement automated scaling mechanisms to dynamically adjust resources based on demand.

- * Reliability:

- * Ensure high availability and fault tolerance of the application and infrastructure.

- * Implement redundant systems and disaster recovery procedures.
- * Monitor system performance and proactively address potential issues.
- * Compliance:
 - * Adhere to all relevant financial regulations and industry best practices.
 - * Maintain detailed documentation of security policies and procedures.
 - * Conduct regular compliance audits and certifications.
- * Usability:
 - * Provide a user-friendly and efficient remote access solution using MobaXterm.
 - * Streamline deployment and maintenance processes.
 - * Implement effective monitoring and alerting systems.

By achieving these objectives, the financial institution will be able to securely host its internal application, protect sensitive client data, and maintain compliance with regulatory requirements.

Project Scope:

Infrastructure Setup:

- * VPC Creation:
 - * Design and implement a secure VPC with appropriate security groups and network access control lists (ACLs).
 - * Configure routing tables and internet gateways to enable communication between EC2 instances and the internet.
- * EC2 Instance Provisioning:
 - * Provision EC2 instances with appropriate specifications (CPU, memory, storage) to host the internal application.
 - * Install and configure the operating system and necessary software (e.g., web server, application server, database).
 - * Implement security hardening measures, including:
 - * Disabling unnecessary services and ports.
 - * Applying security patches and updates.
 - * Configuring strong password policies.
- * MobaXterm Integration:

- * Configure MobaXterm for secure SSH access to EC2 instances.
- * Implement role-based access control to limit user privileges.
- * Configure MobaXterm to enforce security best practices, such as strong password policies and two-factor authentication.

Application Deployment and Configuration:

* Application Deployment:

- * Deploy the internal application to the EC2 instances.
- * Configure the application to access necessary databases and other services.
- * Implement load balancing and failover mechanisms to ensure high availability.

* Security Configuration:

- * Configure the application to enforce strong security measures, including:
 - * Input validation and sanitization.
 - * Secure coding practices.
 - * Regular security testing and vulnerability scanning.
- * Implement data encryption at rest and in transit.
- * Configure secure session timeouts and password expiration policies.

Monitoring and Maintenance:

* Monitoring:

- * Implement monitoring tools to track system performance and identify potential issues.
- * Configure alerts for critical events, such as system failures or security breaches.

* Maintenance:

- * Perform regular system maintenance, including:
 - * Patching and updating software.
 - * Backing up data.
 - * Security audits and vulnerability assessments.

* Incident Response:

- * Develop and test an incident response plan.
- * Have procedures in place to quickly identify, contain, and resolve security incidents.

Compliance and Security:

- * Compliance:

- * Ensure compliance with relevant industry standards and regulations (e.g., PCI DSS, HIPAA).

- * Conduct regular security audits and certifications.

- * Maintain detailed documentation of security policies and procedures.

- * Security Best Practices:

- * Implement strong password policies and multi-factor authentication.

- * Regularly review and update security policies and procedures.

- * Conduct security awareness training for all personnel.

By following this scope, the project will deliver a secure, scalable, and compliant internal application hosting environment for the financial institution.

Project Timeline

Phase 1: Planning and Design (2 weeks)

- * Define project scope, objectives, and deliverables.

- * Conduct a security risk assessment.

- * Design the VPC architecture, including security groups and network ACLs.

- * Select EC2 instance types and configurations.

- * Develop a deployment and configuration plan.

- * Create a security policy and procedures document.

Phase 2: Infrastructure Setup (2 weeks)

- * Create the VPC and subnets.

- * Configure security groups and network ACLs.

- * Provision EC2 instances and install the operating system.

- * Configure MobaXterm for secure remote access.

- * Implement baseline security measures (e.g., firewall rules, password policies).

Phase 3: Application Deployment and Configuration (2 weeks)

- * Deploy the application to the EC2 instances.

- * Configure the application for optimal performance and security.
- * Implement database security measures (e.g., encryption, access controls).
- * Test the application functionality and performance.

Phase 4: Security Testing and Validation (1 week)

- * Conduct vulnerability scanning and penetration testing.
- * Review security logs and identify potential security risks.
- * Implement any necessary security fixes or configuration changes.

Phase 5: User Acceptance Testing (1 week)

- * Conduct user acceptance testing to verify functionality and usability.
- * Address any issues identified during testing.

Phase 6: Deployment and Go-Live (1 week)

- * Deploy the application to the production environment.
- * Monitor the application closely during the initial period.
- * Conduct post-deployment security review.

Total Project Duration: 9 weeks

Note: This is a general timeline and may vary based on the complexity of the application, the size of the infrastructure, and the specific requirements of the financial institution. It's important to factor in additional time for unforeseen challenges and iterative development.

Budget for Secure Internal Application Hosting

Disclaimer: The following budget is a rough estimate and can vary significantly based on specific requirements, AWS region, instance types, and additional services. It's recommended to use AWS Cost Explorer or consult with an AWS Solutions Architect for a more accurate cost estimate.

Required Resources:

Hardware Resources:

- * EC2 Instances:
 - * Sufficient CPU, memory, and storage to handle the application's workload.
 - * Consider using instance types optimized for specific workloads (e.g., compute-optimized, memory-optimized).
- * EBS Volumes:
 - * Sufficient storage for the application's data and logs.

- * Consider using SSD volumes for better performance.

Software Resources:

* Operating System:

- * A secure and stable operating system (e.g., Linux, Windows Server).

* Application Server:

- * A suitable application server to deploy the application (e.g., Tomcat, JBoss, IIS).

* Database Server:

- * A database server to store the application's data (e.g., MySQL, PostgreSQL, SQL Server).

* MobaXterm:

- * Licenses for authorized personnel to access EC2 instances securely.

* Security Tools:

- * Security tools for vulnerability scanning, penetration testing, and intrusion detection.
- * Encryption tools to protect sensitive data.
- * Logging and monitoring tools to track system activity and identify potential threats.

Human Resources:

* System Administrator:

- * To manage the infrastructure, deploy the application, and perform maintenance tasks.

* Security Engineer:

- * To implement and maintain security measures, conduct security assessments, and respond to security incidents.

* Developer/Engineer:

- * To develop, test, and deploy the application.

Network Resources:

- * VPC: A virtual private cloud to isolate the application environment.

- * Subnets: To segment the network into different functional areas (e.g., public, private).

- * Security Groups: To control inbound and outbound traffic to EC2 instances.

- * Network ACLs: To filter traffic at the subnet level.
- * Internet Gateway: To enable internet connectivity for the EC2 instances.
- * Route Tables: To define how traffic is routed within the VPC.

By carefully planning and allocating these resources, the financial institution can ensure the successful deployment and operation of the secure internal application.

Scenario 2:

High-Availability WebHosting for an E-Commerce Platform

An e-commerce company requires a high-availability solution to host its website, which experiences significant traffic fluctuations. The project uses multiple EC2 instances spread across different availability zones within a VPC to ensure fault tolerance and load balancing.

MobaXterm facilitates the quick deployment of updates and management of these instances, ensuring minimal downtime. This setup allows the e-commerce platform to handle peak traffic periods efficiently while maintaining consistent performance and security.

Project objectives:

Project Objective for High-Availability Web Hosting for an E-commerce Platform

To establish a highly available, scalable, and secure web hosting solution for an e-commerce platform, leveraging AWS EC2 instances across multiple Availability Zones within a VPC, and utilizing MobaXterm for efficient remote management. This solution will ensure minimal downtime, optimal performance, and enhanced security.

Key Objectives:

- * High Availability:
 - * Implement a redundant infrastructure to minimize downtime and service disruptions.
 - * Utilize load balancing to distribute traffic across multiple EC2 instances.
 - * Implement automatic failover mechanisms to switch to redundant resources in case of failures.
- * Scalability:
 - * Design the infrastructure to handle increased traffic and growth.

- * Implement auto-scaling capabilities to automatically adjust resources based on demand.
- * Security:
 - * Implement robust security measures to protect the platform and customer data, including:
 - * Strong access controls and authentication mechanisms.
 - * Encryption of data both at rest and in transit.
 - * Regular security audits and vulnerability assessments.
 - * Web application firewalls (WAF) to protect against web attacks.
- * Performance:
 - * Optimize the application and infrastructure for performance, including:
 - * Caching mechanisms to reduce server load.
 - * Content Delivery Networks (CDNs) to improve website load times.
 - * Database optimization techniques.
- * Manageability:
 - * Utilize MobaXterm for efficient remote access and management of EC2 instances.
 - * Implement automated deployment and configuration management tools.
 - * Establish effective monitoring and alerting systems.

By achieving these objectives, the e-commerce platform will be able to deliver a seamless user experience, maintain business continuity, and protect sensitive customer data.

Project Scope:

Infrastructure Setup:

- * VPC Creation:
 - * Design and implement a highly available VPC with multiple Availability Zones.
 - * Configure security groups and network ACLs to secure the VPC.
- * EC2 Instance Provisioning:
 - * Provision multiple EC2 instances across different Availability Zones.
 - * Configure the EC2 instances with the required operating system and software (e.g., web server, application server, database).
 - * Implement security hardening measures, such as:

- * Disabling unnecessary services and ports.
- * Applying security patches and updates.
- * Configuring strong password policies.
- * Load Balancing:
 - * Configure an Elastic Load Balancer to distribute traffic across multiple EC2 instances.
 - * Implement health checks to monitor instance health and automatically route traffic to healthy instances.
- * MobaXterm Integration:
 - * Configure MobaXterm for secure SSH access to EC2 instances.
 - * Implement role-based access control to limit user privileges.
 - * Configure MobaXterm to enforce security best practices, such as strong password policies and two-factor authentication.

Application Deployment and Configuration:

- * Application Deployment:
 - * Deploy the e-commerce application on the EC2 instances.
 - * Configure the application to access the database and other necessary services.
 - * Implement load balancing and failover mechanisms to ensure high availability.
- * Security Configuration:
 - * Configure the application to enforce strong security measures, including:
 - * Input validation and sanitization.
 - * Secure coding practices.
 - * Regular security testing and vulnerability scanning.
 - * Implement data encryption at rest and in transit.
 - * Configure secure session timeouts and password expiration policies.

Monitoring and Maintenance:

- * Monitoring:
 - * Implement monitoring tools to track system performance and identify potential issues.
 - * Configure alerts for critical events, such as system failures or security breaches.

- * Maintenance:

- * Perform regular system maintenance, including:

- * Patching and updating software.

- * Backing up data.

- * Security audits and vulnerability assessments.

- * Incident Response:

- * Develop and test an incident response plan.

- * Have procedures in place to quickly identify, contain, and resolve security incidents.

Security and Compliance:

- * Security Best Practices:

- * Implement strong password policies and multi-factor authentication.

- * Regularly review and update security policies and procedures.

- * Conduct security awareness training for all personnel.

- * Compliance:

- * Ensure compliance with relevant industry standards and regulations (e.g., PCI DSS, HIPAA).

- * Conduct regular security audits and certifications.

- * Maintain detailed documentation of security policies and procedures.

Project Timeline

Phase 1: Planning and Design (2 weeks)

- * Define project scope, objectives, and deliverables.

- * Conduct a security risk assessment.

- * Design the VPC architecture, including security groups and network ACLs.

- * Select EC2 instance types and configurations.

- * Develop a deployment and configuration plan.

- * Create a security policy and procedures document.

Phase 2: Infrastructure Setup (2 weeks)

- * Create the VPC and subnets.

- * Configure security groups and network ACLs.
- * Provision EC2 instances and install the operating system.
- * Configure MobaXterm for secure remote access.
- * Implement baseline security measures (e.g., firewall rules, password policies).

Phase 3: Load Balancer Configuration (1 week)

- * Configure Elastic Load Balancer and health checks.

Phase 4: Application Deployment and Configuration (2 weeks)

- * Deploy the e-commerce application to the EC2 instances.
- * Configure the application for optimal performance and security.
- * Implement database security measures (e.g., encryption, access controls).
- * Test the application functionality and performance.

Phase 5: Testing and Optimization (2 weeks)

- * Conduct performance and security testing, and optimize configuration.

Phase 6: Security Testing and Validation (1 week)

- * Conduct vulnerability scanning and penetration testing.
- * Review security logs and identify potential security risks.
- * Implement any necessary security fixes or configuration changes.

Phase 7: Go-Live and Monitoring (1 week)

- * Deploy the application to the production environment.
- * Monitor the application closely during the initial period.
- * Conduct post-deployment security review.

Required Resources:

Hardware Resources:

- * EC2 Instances:
 - * Sufficient CPU, memory, and storage to handle the application's workload.
 - * Consider using instance types optimized for specific workloads (e.g., compute-optimized, memory-optimized).

* EBS Volumes:

- * Sufficient storage for the application's data and logs.
- * Consider using SSD volumes for better performance.

Software Resources:

* Operating System:

- * A secure and stable operating system (e.g., Linux, Windows Server).

* Web Server:

- * A suitable web server to serve the application (e.g., Apache, Nginx).

* Application Server:

- * An application server to run the e-commerce application (e.g., Tomcat, JBoss, Node.js).

* Database Server:

- * A database server to store the application's data (e.g., MySQL, PostgreSQL, Amazon RDS).

* MobaXterm:

- * Licenses for authorized personnel to access EC2 instances securely.

* Security Tools:

- * Security tools for vulnerability scanning, penetration testing, and intrusion detection.
- * Encryption tools to protect sensitive data.
- * Logging and monitoring tools to track system activity and identify potential threats.

Network Resources:

- * VPC: A virtual private cloud to isolate the application environment.
- * Subnets: To segment the network into different functional areas (e.g., public, private).
- * Security Groups: To control inbound and outbound traffic to EC2 instances.
- * Network ACLs: To filter traffic at the subnet level.
- * Internet Gateway: To enable internet connectivity for the EC2 instances.
- * Route Tables: To define how traffic is routed within the VPC.

Human Resources:

- * System Administrator:

- * To manage the infrastructure, deploy the application, and perform maintenance tasks.

- * Network Engineer:

- * To design and configure the network infrastructure.

- * Developer/Engineer:

- * To develop, test, and deploy the e-commerce application.

- * Security Engineer:

- * To implement and maintain security measures, conduct security assessments, and respond to security incidents.

By carefully planning and allocating these resources, the e-commerce company can ensure the successful deployment and operation of the high-availability web hosting solution.

Scenario 3:

Custom Web Application Deployment for a Start-Up

A tech start-up needs to deploy a custom web application that will serve as its primary business platform. The project sets up an EC2 instance within a VPC to host the application, ensuring it is secure and isolated from other internet traffic. MobaXterm is used to streamline the deployment process and manage the application server. This scenario provides the start-up with a flexible, scalable, and cost-effective solution, allowing them to focus on application development and business growth while maintaining control over their hosting environment.

Project Objectives:

- * Secure and Reliable Infrastructure:

- * Establish a secure and reliable EC2 instance within a VPC to host the web application.

- * Implement appropriate security measures, such as firewalls and access controls, to protect the application and its data.

- * Ensure high availability and redundancy by configuring appropriate infrastructure settings.

- * Efficient Deployment and Management:

- * Utilize MobaXterm to streamline the deployment process and simplify server management tasks.

- * Automate repetitive tasks to reduce manual effort and minimize human error.

- * Implement a robust monitoring system to track application performance and proactively identify issues.

- * Scalability and Performance Optimization:

- * Design the infrastructure to accommodate future growth and increased traffic.

- * Optimize the application and server configuration for optimal performance and resource utilization.

- * Implement load balancing and auto-scaling mechanisms to handle fluctuating workloads.

Secondary Objectives:

- * Cost-Effective Solution:

- * Select appropriate EC2 instance types and configurations to balance performance and cost.

- * Leverage AWS cost optimization tools and strategies to minimize expenses.

- * Compliance and Security Adherence:

- * Ensure compliance with relevant industry standards and regulations (e.g., GDPR, HIPAA).

- * Implement regular security audits and vulnerability assessments to identify and address security risks.

By achieving these objectives, the start-up will have a robust, scalable, and secure web application platform that supports its business growth and innovation.

Project Scope:

1. Infrastructure Setup:

- * Provision an EC2 instance within a dedicated VPC.

- * Configure the EC2 instance with appropriate operating system and necessary software (e.g., web server, database, application server).

- * Implement security measures, including firewall rules, access controls, and security group configurations.

2. Application Deployment:

- * Deploy the web application to the EC2 instance using MobaXterm.

- * Configure the application server (e.g., Apache, Nginx) to serve the web application.

- * Set up the database (e.g., MySQL, PostgreSQL) and configure the application to interact with it.

3. Network Configuration:

- * Configure network settings (e.g., DNS, routing) to ensure the application is accessible.
- * Implement load balancing (if required) to distribute traffic across multiple instances.

4. Monitoring and Logging:

- * Set up monitoring tools to track application performance and system health.
- * Configure logging to record application and system events for troubleshooting and analysis.

5. Security Best Practices:

- * Implement regular security patches and updates.
- * Conduct security audits and vulnerability assessments.
- * Implement strong password policies and access controls.
- * Consider using a web application firewall (WAF) for additional security.

6. Backup and Disaster Recovery:

- * Establish a backup strategy for the application and database.
- * Implement a disaster recovery plan to minimize downtime in case of failures.

7. Documentation:

- * Create detailed documentation for the infrastructure, application deployment, and configuration.
- * Document security procedures and best practices.

By completing these tasks, the project will deliver a secure, reliable, and scalable web application platform that meets the start-up's needs.

Project Timeline:

Phase 1: Planning and Preparation (1-2 weeks)

- * Project kickoff and team formation
- * Requirements gathering and analysis
- * Technical design and architecture
- * Resource allocation

Phase 2: Infrastructure Setup (1-2 weeks)

- * VPC creation and configuration
- * EC2 instance provisioning and configuration
- * Security group and firewall rule setup
- * Network configuration (DNS, routing)

Phase 3: Application Deployment (1-2 weeks)

- * Application deployment and configuration
- * Database setup and configuration
- * Web server configuration

Phase 4: Testing and Optimization (2-3 weeks)

- * Functional testing
- * Performance testing and optimization
- * Security testing and vulnerability assessment

Phase 5: Deployment and Go-Live (1-2 weeks)

- * Final deployment and configuration
- * User acceptance testing (UAT)
- * Go-live and post-deployment verification

Phase 6: Post-Deployment Activities (1-2 weeks)

- * Monitoring and logging setup
- * Documentation and knowledge transfer
- * Ongoing maintenance and support

Project Resources:

****Development Team**:**

Skilled developers with experience in web application development, including front-end, back-end, and full-stack developers.

****Project Manager**:**

To oversee the project, manage timelines, and ensure communication between stakeholders and the development team.

****Cloud Services**:**

AWS EC2 instances for hosting the application within a VPC, ensuring security and isolation.

****Development Tools**:**

Integrated Development Environment (IDE), version control system (e.g., Git), and collaboration tools (e.g., Slack, Jira).

****Deployment Tools**:**

MobaXterm for streamlining the deployment process and managing the application server.

****Testing Tools**:**

Tools for quality assurance and testing, such as Selenium, JUnit, or other automated testing frameworks.

****Security Tools**:**

Solutions for ensuring application security, such as SSL certificates, firewalls, and regular security audits.

****Documentation**:**

Detailed documentation of the application architecture, codebase, and deployment procedures.

****Maintenance Plan**:**

Ongoing support and maintenance plan to address any issues and implement updates.

Harnessing EC2, VPC & MobaXterm for Cutting-Edge Entity Hosting

Architecture:



Pre-requisites:

1. AWS Account Setup:

https://youtu.be/CjKhQoYeR4Q?si=ui8Bvk_M4FfVM-Dh

2. Understanding of IAM: <https://youtu.be/gsgdAyGhV0o?si=3qg-bULgkD4LXNvR>

3. Knowledge of Amazon EC2: <https://youtu.be/8TlukLu11Yo?si=MUj0nEAOESRhHUIz>

4. MobaXterm : <https://youtu.be/dvoU2SKG6oA?si=Hs8Pu4Crry5-BRrD>

Project Flow:

Project Initialization:

- Define objectives, scope, and KPIs for deploying the Organization's website.
- Set up AWS environment, including EC2 instance configuration and MobaXterm setup.

EC2 Instance Creation:

- Launch an EC2 instance to host the organization's website(acerTIFY).
- Select the appropriate instance type based on resource requirements.

MobaXterm Configuration:

- Install and configure MobaXterm for remote access to the EC2 instance.
- Set up SSH keys and configure session settings for seamless connection.

Website Deployment:

- Transfer website files to the EC2 instance using MobaXterm.
- Configure the web server (e.g., Apache, Nginx) to serve the acerTIFY website.

Testing and Optimization:

- Test the website for functionality, performance, and security.
- Optimize server settings and website configurations for better performance.

Monitoring and Maintenance:

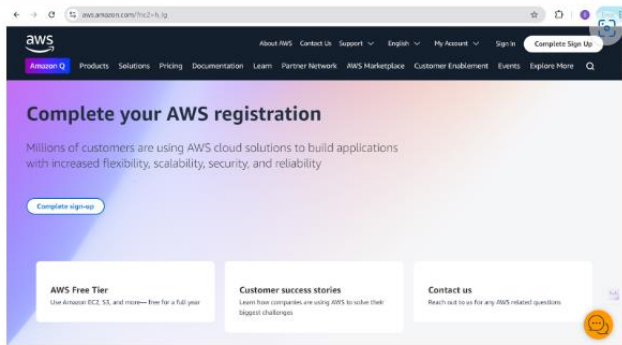
- Implement monitoring tools to track website performance and uptime.

- Regularly update and maintain the website and server to ensure reliability.

Create an AWS account and login to console

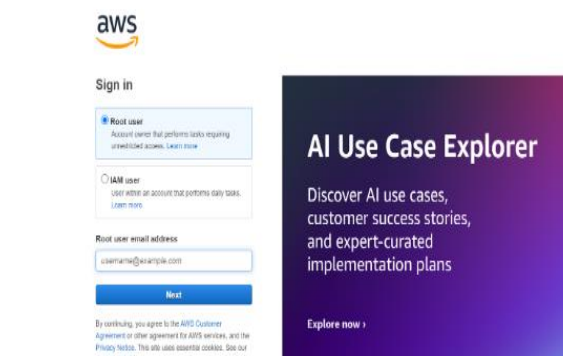
In this milestone, we will create and sign in to the AWS console.

Create AWS Account:



Activity 1.2 : Log in to AWS Management Console:

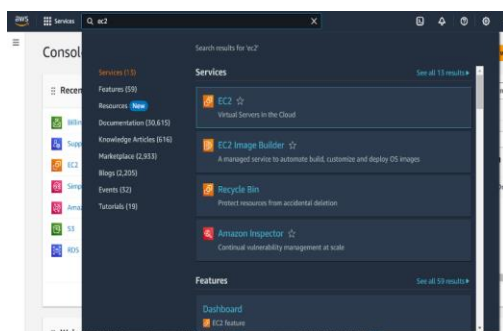
Access the AWS Management Console using your credentials.



Setting Up AWS EC2 Instance

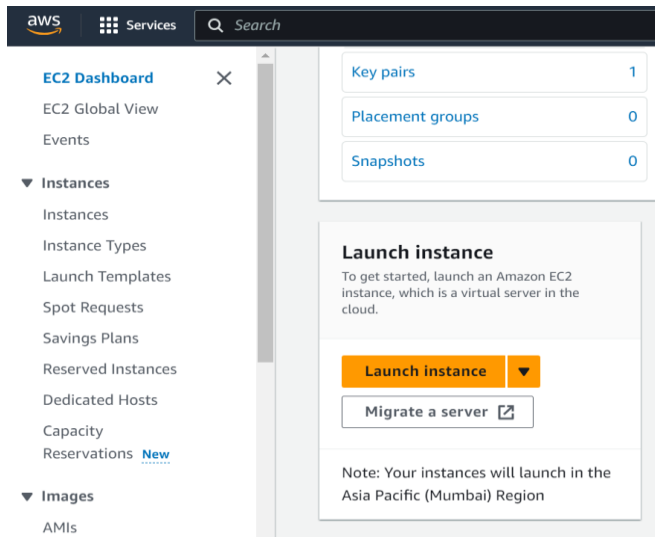
In this Milestone we launch and configure an EC2 instance to host the website. This step involves creating an EC2 instance that will serve as the server environment for the website.

Creating EC2 instance

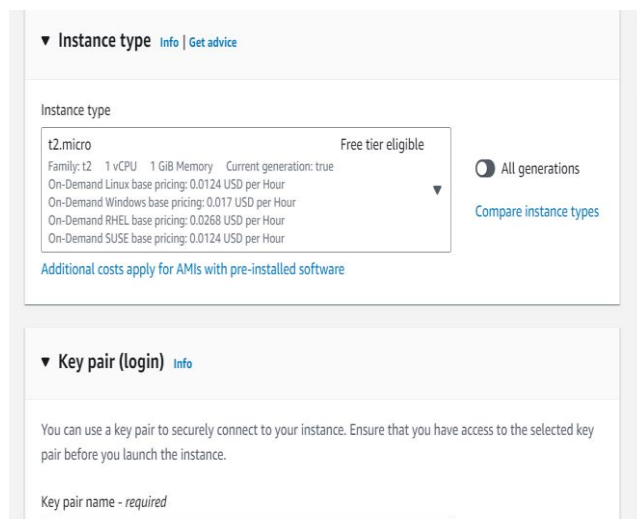


- Search for EC2 in navigation bar
- Click on Ec2

- In the EC2 Dashboard click on Launch Instance



- Click on "Launch Instance" to start the instance creation wizard.
- Choose an Amazon Machine Image (AMI) suitable for your application (e.g., Amazon Linux 2 or Ubuntu).
- Select an instance type (e.g., t2.micro for testing or t3.medium for production).
- Configure instance details, including network settings and storage options.
- Create or select an existing key pair for secure SSH access.



Number of instances ... info

Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

website

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type


☒ RSA
RSA encrypted private and public key pair

☐ ED25519
ED25519 encrypted private and public key pair

Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

 When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. [Learn more](#)

Cancel Create key pair

Storing the key pair as .pem file

Create a key pair and keep the .pem file in the local system.

Network settings

- Allow SSH Traffic from anywhere
- Launch instance

▼ Network settings

Info

Edit

Network

Info

vpc-017027b2b085367dc

Subnet

Info

No preference (Default subnet in any availability zone)

Auto-assign public IP

Info

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)

Info

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group

We'll create a new security group called 'launch-wizard-2' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere

0.0.0.0/0

☐ Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

☐ Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

▼ Configure storage

Info

Advanced

1x

8

GiB

gp3

Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GiB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

Click refresh to view backup information

The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems

Edit

► Advanced details

Info

EC2

> Instances

> Launch an instance

Success

Successfully initiated launch of instance (i-0f27740eaf4b9283a)

► Launch log

Next Steps

Q What would you like to do next with this instance, for example "create alarm" or "create backup"

<

1

2

3

4

5

6

>

Number of instances

Info

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.5.2...read more

ami-02b49a24cfb95941c

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year

Cancel

Launch instance

Review commands

- Review and launch the instance, then wait for the instance status to change to "Running."

Activity 2.5 : Setting up Inbound and Outbound rules

The screenshot displays the AWS Management Console interface for an EC2 instance's security configuration. The top navigation bar includes tabs for Details, Status and alarms, Monitoring, Security (selected), Networking, Storage, and Tags.

Security details:

- IAM Role:** -
- Owner ID:** 345594583920
- Launch time:** Tue Aug 27 2024 11:20:10 GMT+0530 (India Standard Time)
- Security groups:** sg-0236ecef4fe759456 (launch-wizard-2)

Inbound rules:

Name	Security group rule ID	Port range	Protocol	Source
-	sgr-0d492ace97d1c9694	22	TCP	0.0.0.0/0

Edit inbound rules:

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0d492ace97d1c9694	SSH	TCP	22	My IP	
-	HTTP	TCP	80	An...	
-	HTTPS	TCP	443	An...	

Outbound rules:

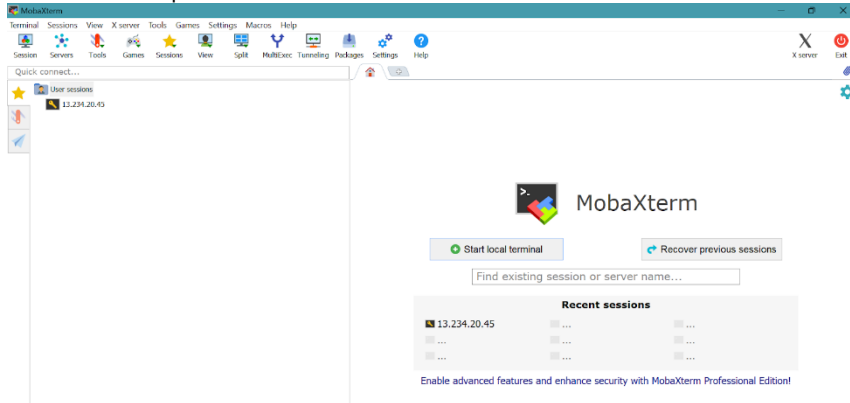
Security group rule...	IP version	Type	Protocol	Port range
sgr-0a2b9f432765ee420	IPv4	All traffic	All	All

Install and Configure MobaXterm

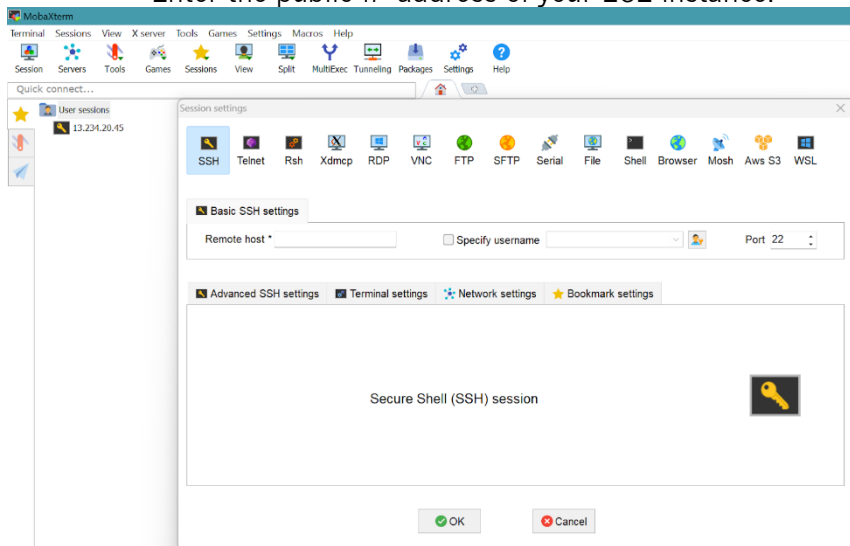
In this milestone we set up MobaXterm to manage remote access to the EC2 instance. MobaXterm is a powerful terminal application that combines various network tools into a single portable executable file.

Mobaxterm

- Download and install MobaXterm from the official website.
- Open MobaXterm and create a new SSH session.



- Enter the public IP address of your EC2 instance.



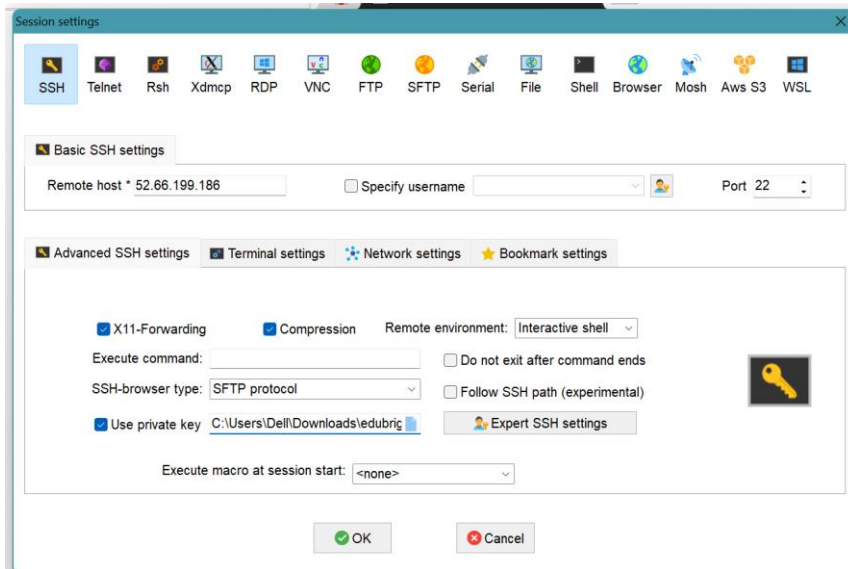
Click on Advanced settings and Upload the .pem file into it

Public IPv4 address

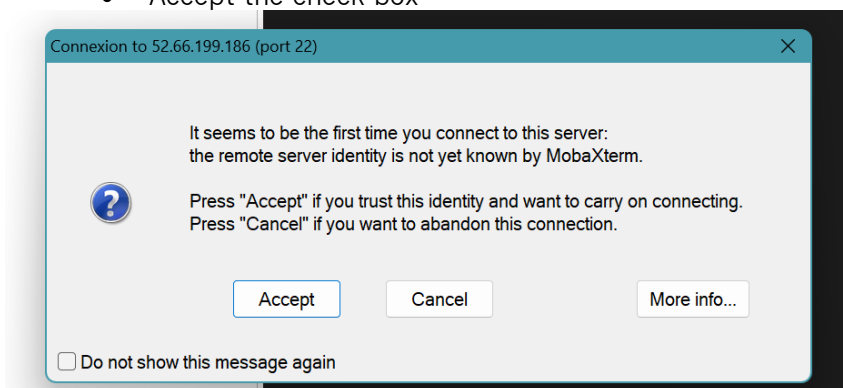
 65.0.94.131 | [open address](#) 



website (1).pem



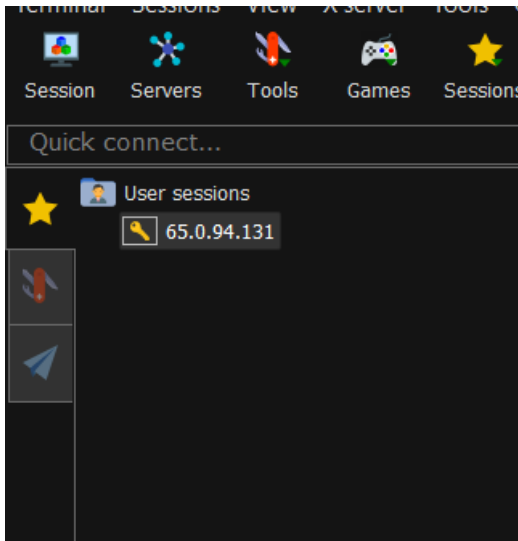
- Accept the check box



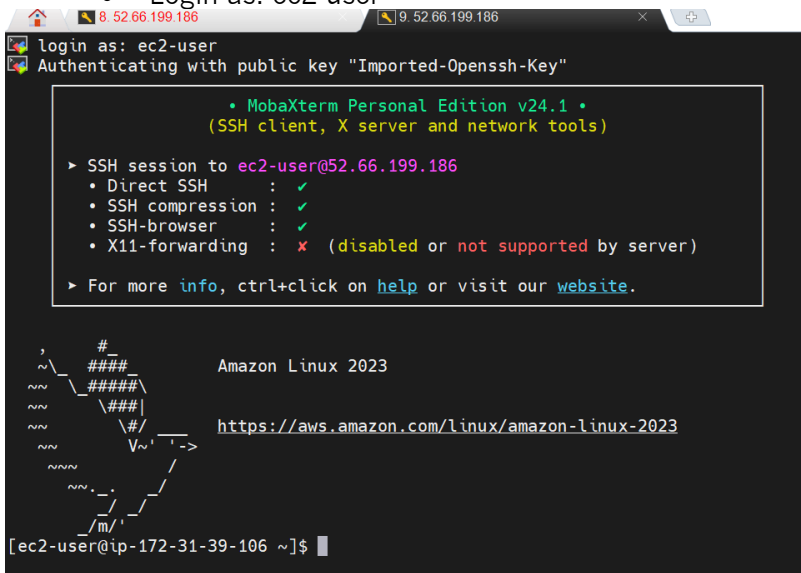
Configure Web Server on EC2

In this Milestone, we set up a web server environment on the EC2 instance to serve the EduBridge website. This includes installing necessary software and configuring the server.

Login into the SSH Session



- Login as: ec2-user



- EC2 Instance is launched and further we need to create a web server and need to upload the files to host the website

Update and Install Web Server Software

- Update package lists with `sudo apt-get update` (Ubuntu) or `sudo yum update` (Amazon Linux).

Install Apache or Nginx:

- For Apache: `sudo apt-get install apache2` (Ubuntu) or `sudo yum install httpd` (Amazon Linux).
- For Nginx: `sudo apt-get install nginx` (Ubuntu) or `sudo yum install nginx` (Amazon Linux).

```
[ec2-user@ip-172-31-39-106 ~]$ sudo yum install httpd -y
Last metadata expiration check: 0:23:00 ago on Tue Aug 27 05:50:54 2024.
Dependencies resolved.
=====
Package                Arch      Version              Repository           Size
=====
Installing:
httpd                  x86_64    2.4.62-1.amzn2023    amazonlinux           48
Installing dependencies:
apr                    x86_64    1.7.2-2.amzn2023.0.2 amazonlinux           129
apr-util               x86_64    1.6.3-1.amzn2023.0.1 amazonlinux           98
generic-logos-httpd    noarch    18.0.0-12.amzn2023.0.3 amazonlinux           19
httpd-core              x86_64    2.4.62-1.amzn2023    amazonlinux           1.4
httpd-filesystem        noarch    2.4.62-1.amzn2023    amazonlinux           14
httpd-tools             x86_64    2.4.62-1.amzn2023    amazonlinux           81
libbrotli               x86_64    1.0.9-4.amzn2023.0.2 amazonlinux           315
mailcap                 noarch    2.1.49-3.amzn2023.0.3 amazonlinux           33
Installing weak dependencies:
apr-util-openssl        x86_64    1.6.3-1.amzn2023.0.1 amazonlinux           17
mod_http2                x86_64    2.0.27-1.amzn2023.0.3 amazonlinux           166
mod_lua                  x86_64    2.4.62-1.amzn2023    amazonlinux           61
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm 258 kB/s | 17 kB 00:00
(2/12): apr-1.7.2-2.amzn2023.0.2.x86_64.rpm 1.8 MB/s | 129 kB 00:00
(3/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm 1.3 MB/s | 98 kB 00:00
(4/12): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.r 1.0 MB/s | 19 kB 00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64.rpm 2.6 MB/s | 48 kB 00:00
(6/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm 35 MB/s | 1.4 MB 00:00
```

To start the services, we need to use the below snippet of code

```
Complete!
[ec2-user@ip-172-31-39-106 ~]$ sudo service httpd start
Redirecting to /bin/systemctl start httpd.service
[ec2-user@ip-172-31-39-106 ~]$ █
```

To know the present working directory

```

[ec2-user@ip-172-31-39-106 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-39-106 ~]$

```

To be worked in the html folder where we need to upload the html files, css files and JS scripts files

```

[ec2-user@ip-172-31-39-106 ~]$ cd /var/www/html
[ec2-user@ip-172-31-39-106 html]$ pwd
/var/www/html
[ec2-user@ip-172-31-39-106 html]$

```

Deploy EduBridge Website Code

In this milestone we upload and deploy the website code to the web server on the EC2 instance. This involves transferring files and configuring the web server to serve the website.

Upload Website Files Using MobaXterm

- Use MobaXterm's SFTP functionality to transfer website files to the EC2 instance.
- Navigate to the /var/www/html directory (or the relevant directory for Nginx) and upload the files.

```

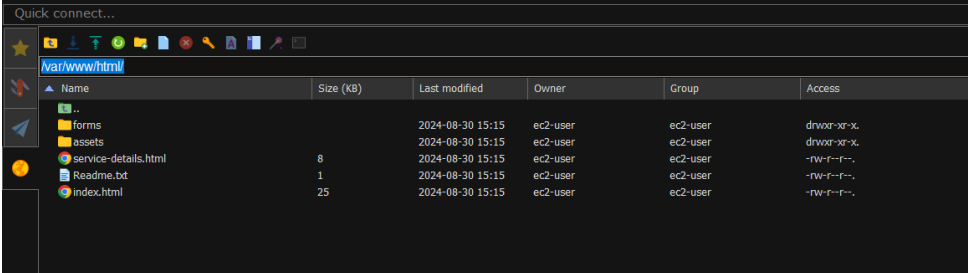
[ec2-user@ip-172-31-39-106 ~]$ cd /var/www/html
[ec2-user@ip-172-31-39-106 html]$ pwd
/var/www/html
[ec2-user@ip-172-31-39-106 html]$

```

```

/var/www/html
[ec2-user@ip-172-31-39-106 html]$ sudo chmod 777 /var/www/html
[ec2-user@ip-172-31-39-106 html]$

```



Name	Size (KB)	Last modified	Owner	Group	Access
forms		2024-08-30 15:15	ec2-user	ec2-user	drwxr-xr-x
assets		2024-08-30 15:15	ec2-user	ec2-user	drwxr-xr-x
service-details.html	8	2024-08-30 15:15	ec2-user	ec2-user	-rw-r--r--
Readme.txt	1	2024-08-30 15:15	ec2-user	ec2-user	-rw-r--r--
index.html	25	2024-08-30 15:15	ec2-user	ec2-user	-rw-r--r--

Configure Web Server Directory and Permissions

- Verify that the web server points to the correct directory where files are uploaded.
- Set appropriate file permissions: `sudo chmod 777 /var/www/html`.

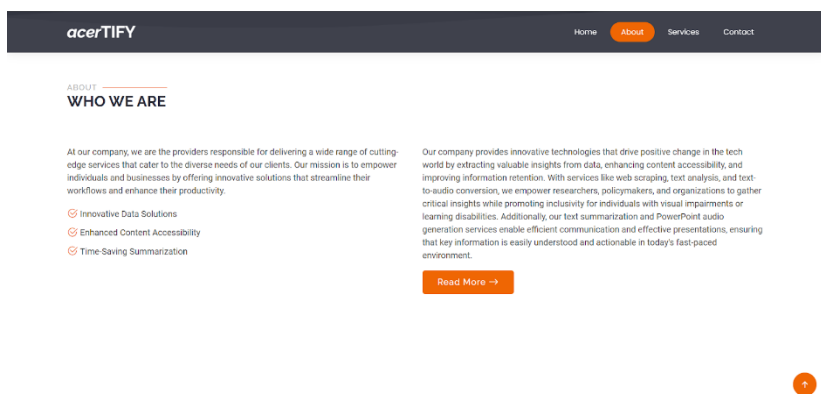
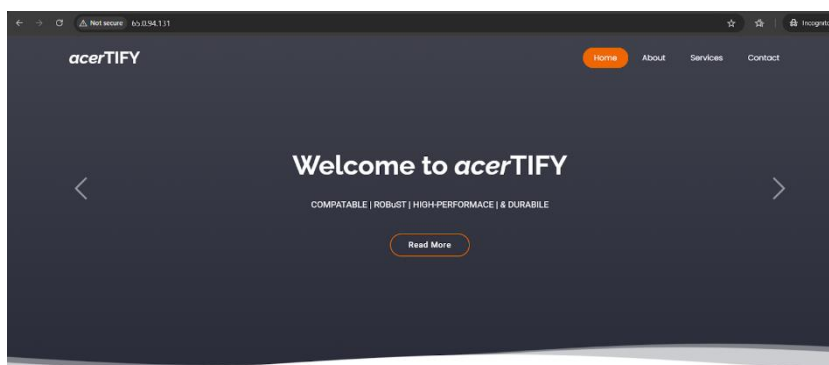
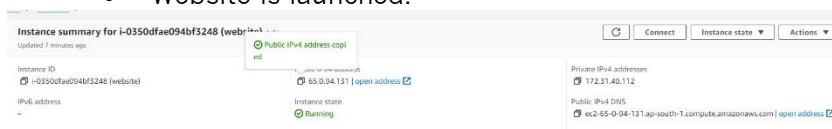
```
[ec2-user@ip-172-31-39-106 html]$ sudo chmod 777 /var/www/html
[ec2-user@ip-172-31-39-106 html]$ sudo chmod 777 index.html
[ec2-user@ip-172-31-39-106 html]$
```

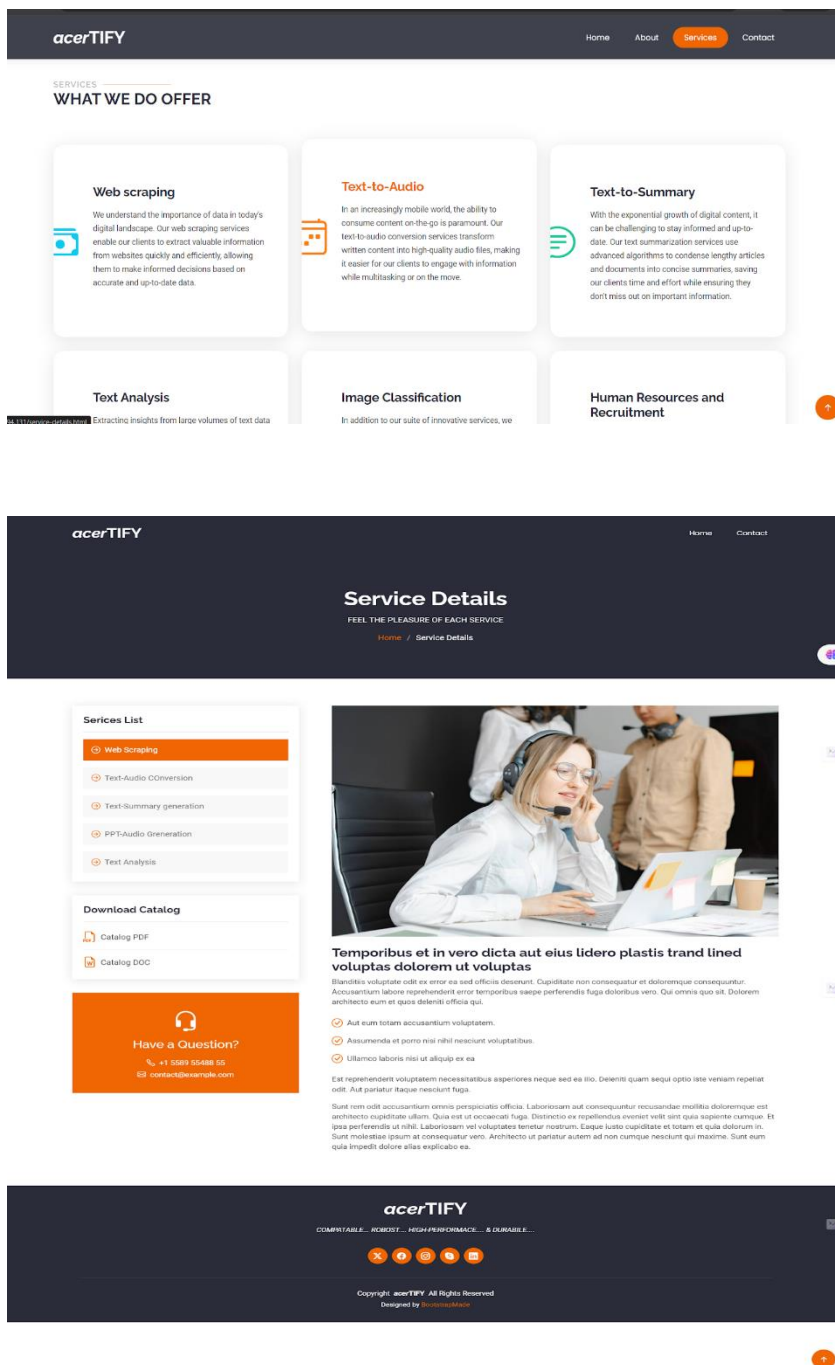
Launching the website

In this milestone, we launch the website in the ec2 dashboard.

hosting website

- Go to EC2 instance.
- Click on the IPv4 address.
- Website is launched.





Monitor and Optimize Performance

- Performance and efficiency by implementing real-time monitoring with Amazon CloudWatch and proactive alerts.
- It ensures dynamic resource management through Elastic Load Balancing and Auto Scaling, optimizing traffic distribution and resource usage. Additionally, MobaXterm facilitates efficient server management, while Amazon RDS Performance Insights (if applicable) monitors and optimizes database performance.

- These combined strategies deliver a robust, scalable, and cost-effective hosting solution for your organization's entity.

Results:

Scenario 1: Secure Internal Application Hosting for a Financial Institution

The financial institution successfully hosted a secure internal application that handles sensitive client data by using EC2 instances within a VPC. This setup created isolated environments for hosting the application, ensuring security and compliance with financial regulations. MobaXterm was used for secure remote access and management, allowing only authorized personnel to deploy updates or perform maintenance.

Scenario 2: High-Availability Web Hosting for an E-Commerce Platform

The e-commerce company achieved high availability and fault tolerance by utilizing multiple EC2 instances spread across different availability zones within a VPC. This setup ensured load balancing and minimal downtime, even during significant traffic fluctuations. MobaXterm facilitated quick deployment of updates and management of these instances, enabling the platform to efficiently handle peak traffic periods while maintaining consistent performance and security.

Scenario 3: Custom Web Application Deployment for a Start-Up

The tech start-up successfully deployed a custom web application as its primary business platform by setting up an EC2 instance within a VPC. This ensured the application was secure and isolated from other internet traffic. MobaXterm streamlined the deployment process and managed the application server, providing the start-up with a flexible, scalable, and cost-effective solution. This allowed the start-up to focus on application development and business growth while maintaining control over their hosting environment.

Conclusion:

"Harnessing EC2, VPC, and MobaXterm for Cutting-Edge Entity Hosting" represents a sophisticated approach to modern web hosting, integrating advanced cloud technologies with efficient management tools. By employing proactive monitoring, dynamic resource allocation, and streamlined server management, this project ensures optimal performance, scalability, and security. This solution not only meets the immediate hosting needs of the organization but also provides a resilient foundation for future growth and innovation.