**Cloud Server Project Report**

**Cloud Server Project & Video Explainer**

**ICT171 Assignment 2 – 2025**



**Student ID:** 35435334

**Global IP Address:** **13.239.21.33**

# 1. Introduction

This report outlines the process of setting up a cloud-based server using Infrastructure as a Service (IaaS). The server was deployed on **Amazon EC2**, and it serves as a platform for the **Eventify** web application. **Eventify** is a cloud-based web service designed to streamline the creation, registration, and management of academic events such as workshops, seminars, and cultural events.

I have used **AWS EC2** for hosting the server and configured it to be accessible via the public IP **13.239.21.33**. This document serves as a guide for replicating the server setup and for understanding the steps taken to deploy the application.

# 2. Server Setup and Configuration

## 2.1. Creating the Cloud Instance

1. **Select the Cloud Provider:**

For this project, I used **AWS EC2**, which offers **Infrastructure as a Service (IaaS)**, allowing us to create virtual machines (VMs) to host our server. EC2 instances provide scalable compute capacity in the cloud, and this setup allows easy expansion of the application as needed.

1. **Creating the EC2 Instance**:

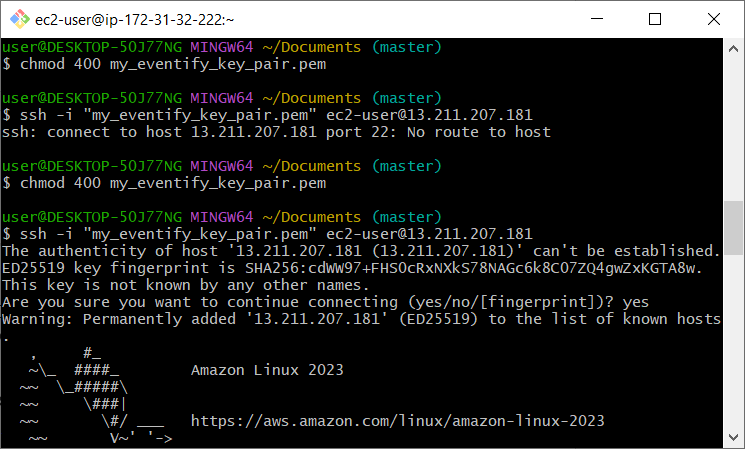
I created an EC2 instance using the following configuration:

* + - **Instance Type**: **t2.micro** (this is a cost-effective instance type that falls under the AWS Free Tier)
    - **OS**: **Amazon Linux 2** (default AMI for Amazon EC2, which uses **ec2-user** as the default SSH user)
* **Security Group**: A custom security group was configured to allow the following inbound rules:
  + - **HTTP** (port 80): To allow web traffic.
    - **SSH** (port 22): To allow secure access for server management.

1. **SSH Access:**

To access the server, I used **SSH**. Here’s the command for SSH access:

$ ssh -i "my\_eventify\_key\_pair.pem" ec2-user@13.211.207.181



## 2.2. Installing and Configuring the Web Server

**1.Updating the Server:**

* I began by updating the server's package repository to ensure everything was up to date:

**sudo yum update -y**

**2.Installing Git:**

* Next, I installed Git to clone the Eventify repository from GitHub:

**sudo yum install git -y**

**3. Cloning the Eventify Repository:**

* I cloned the Eventify GitHub repository to the EC2 instance:

**git clone** [**https://github.com/dekhl/eventify.git**](https://github.com/dekhl/eventify.git)

**4.** **Installing Apache HTTP Server:**

* To serve the web application, I installed Apache HTTP Server (httpd):

**sudo yum install httpd -y**

**5. Starting Apache HTTP Server:**

* I started the httpd service to enable the web server:

**sudo systemctl start httpd**

**6.** **Security Group Configuration:**

I added port 80 to the security group to allow additional access:

* In the EC2 Dashboard > Security Groups, a new rule was added for Port 80 (HTTP).

**7. Accessing the Website:**

* The web application can now be accessed via:

http://13.239.21.33

# 3. Configuring the Web Application

Once the server was set up, the **Eventify** web application was configured to run on the EC2 instance. The following steps were as follows:

1. **Configuring the Application Files**:

* After cloning the repository, I navigated the **eventify** folder on the server and ensured all dependencies for the web application were installed.

1. **Testing the Application**:

* Once the application files were set up, I accessed the website to confirm that the **Eventify** application was being served correctly via the Apache server. The website can be viewed by navigating to [**http://13.239.21.33**](http://13.239.21.33).

# 4. Deployment and Testing

After setting up the server and web application, the following testing procedures were carried out:

1. **Testing Connectivity**:

* I tested the accessibility of the website from different browsers and devices to ensure cross-browser compatibility.

1. **Performance Testing**:

* To ensure the server was handling requests effectively, I conducted basic load testing using tools like **Apache Bench** (ab) to test the responsiveness of the website.

1. **Security Testing**:

* The server's security was also checked by attempting to access restricted areas and ensuring that only the necessary ports (HTTP, SSH, and Port 90) were open in the security group.

# 5. Conclusion

* The server setup process was successfully completed with the deployment of **Eventify** on an EC2 instance. The application was configured to run smoothly on **Apache HTTP Server** and can be accessed via the public IP **13.239.21.33**.
* The EC2 instance is scalable, and further enhancements can be made as the **Eventify** platform grows. Future steps include adding more security features such as SSL certificates, enabling automatic scaling, and optimizing server performance for larger user bases.

# 6. References

* AWS EC2 Documentation: <https://aws.amazon.com/documentation/ec2/>
* Apache HTTP Server Documentation: <https://httpd.apache.org/docs/>
* GitHub: <https://github.com/>

# 7. Appendices

**Appendix A**: https://github.com/dekhl/eventify  
**Appendix B**: https://youtu.be/owMHAtA9HVE?si=GtyRbgjkJ5lPISh0