CLOUD SERVER PROJECT REPORT

Project Title: CharityDonation Web

Application Deployment

Student Name: Yasin Arafat

Student Number: 35017278

CONTENTS

Github Repository: yasirarafat100/CharityDonation

Website Link: https://18.139.115.202/

DNS: www.yasinarafat.link or yasinarafat.link

INTRODUCTION

The report provides a detailed description of how the project named "CharityDonation" was planned, implemented and documented for cloud hosting. Amazon Web Services (AWS) Elastic Compute Cloud (EC2) is used as the IaaS provider for the deployment. Our goal with this server is to deliver a static website application that spreads awareness of charity and its benefits to the public.

The work consisted of installing a Linux server, setting up the web application, connecting to GitHub for version control management, configuring access rules and security, optional setup of a DNS server and writing scripts to automate server installation. This assignment let me use what I learned in ICT171 by working on cloud infrastructure and creating professional documentation.

PROJECT DESCRIPTION

My project CharityDonation is a very simple website for the children wellbeing. Our goal is to connect people from around the to help for the wellbeing of the children though donation or as a volunteer.

SERVER PLANNING AND ARCHITECTURE

The initial stage was setting up a basic architectural plan. The system would use an EC2 instance on AWS to host the server, running Amazon Linux 2. The AWS Free Tier made a t2.micro instance the most readily available option. The services within a stack are:

Operating System

Ubuntu

Web Server

Apache HTTP Server

Source Control

GitHub

Cloud Solution

AWS (EC2)

Script Automation

Bash

SERVER DEPLOYMENT STEPS

LAUNCHING EC2 INSTANCE

1. Logged into AWS Console and navigated to EC2 Dashboard.
2. Created a new t2.micro EC2 instance.
3. Selected Amazon Linux 2 AMI (Free Tier Eligible).
4. Configured network settings and created a new security group.
5. Enabled ports 22 and 80.
6. Downloaded PEM key file for SSH access.

INSTALL APACHE AND GIT

sudo apt update -y

sudo apt install httpd git -y

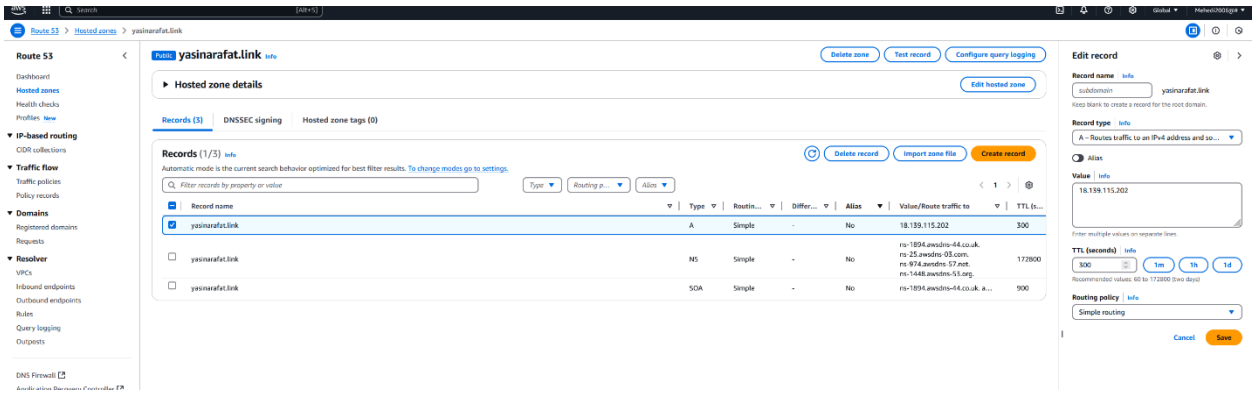
sudo systemctl start httpd

sudo systemctl enable httpd

CREATE DNS RECORD

We are going to use Amazon EC2 TO create DNS using following steps-

1. Login to AWS EC2
2. Open the AWS management console and move to Route 53 service
3. Click Hosted zones
4. Then click on the create hosted zones
5. Now write the domain name you want and choose public hosted zone and finally click create hosted zone
6. Now select the record type base. For example, we are choosing A record.
7. Then in the value field add the public ip address of your server
8. Then set the TTL record or you can leave that on default value
9. Click on create button to save the DNS record



It might take some time for the DNS propagate.

GITHUB INTEGRATION

A public repository was created to host the project files:

- **Repository**

  CharityDonation

- **Files**

  index.html, styles.css, script.js and deployment.js

The repository was cloned directly into the EC2 instance:

cd /var/www/html

sudo rm -rf *

sudo git clone https://github.com/yasirarafat100/CharityDonation.git .

sudo chmod -R 755 /var/www/html

This command ensured the website files were available to the Apache web server.

AUTOMATION SCRIPT

A Bash script was developed to automate server provisioning

#!/bin/bash

sudo yum update -y

sudo yum install httpd git -y

sudo systemctl start httpd

sudo systemctl enable httpd

cd /var/www/html

sudo rm -rf *

sudo git clone https://github.com/yasirarfat100/CharityDonation.git .

sudo chmod -R 755 /var/www/html

This script reduces setup time and ensures a consistent deployment process.

```
Get:15 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Fetched 1015 kB in 2s (649 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
45 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.6).
0 upgraded, 0 newly installed, 0 to remove and 45 not upgraded.
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-25-40:~$ sudo systemctl start apache2
sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-25-40:~$ cd /var/www/html
sudo rm -rf *
```

HTTPS CONFIGURATION AND SSL/TLS SETUP (WITHOUT SCRIPT INCLUSION)

To ensure the safety and proper functioning of a public-facing web application, HTTPS was set up manually on the CharityDonation EC2 server using a self-signed SSL certificate. This means every communication between a user's browser and the server is safe using encryption, regardless of using a registered domain name.

HERE IS HOW I DONE IT

EC2 SECURITY GROUP CONFIGURATION

The first step involved updating the **EC2 instance's security group** to allow inbound traffic on **port 443**, which is used for HTTPS. This was done via the AWS Management Console:

- Navigated to **EC2 > Security Groups**
- Selected the relevant security group attached to the instance
- Edited **Inbound Rules**
- Added:

- **Type:** HTTPS
- **Port Range:** 443
- **Source:** 0.0.0.0/0 (for public access)

INSTALL APACHE AND OPENSSL

After connecting to the EC2 instance via SSH, Apache web server and OpenSSL were installed:

- sudo apt update
- sudo apt install apache2 openssl -y

CREATE SSL DIRECTORY

A secure directory was created to store the self-signed certificate and private key:

- sudo mkdir /etc/apache2/ssl

GENERATE A SELF-SIGNED SSL CERTIFICATE

An SSL certificate valid for 365 days was generated using OpenSSL. The Common Name (CN) used was the public IP address 18.139.115.202, since no domain name was configured.

CONFIGURE APACHE FOR SSL

The default SSL site configuration file (/etc/apache2/sites-available/default-ssl.conf) was updated:

- The SSLCertificateFile and SSLCertificateKeyFile paths were changed to point to the new self-signed certificate and key.
- SSL module was enabled using sudo a2enmod ssl.
- The default SSL site was activated using sudo a2ensite default-ssl.

RESTART APACHE

Apache was restarted to apply the changes:

- sudo systemctl restart apache2

BROWSER ACCESS VERIFICATION

After configuration, the website was successfully accessed using:

https://18.139.115.202

TESTING AND VALIDATION

Post-deployment, validation steps included:

- Accessing the website via browser: https://18.139.115.202/
- Verifying Apache status: sudo systemctl status httpd
- Confirming GitHub files were loaded correctly

Functional testing confirmed that the HTML, CSS, and JS were rendered as expected and the browser could access the site globally.

Our Work

Three pillars of impact: Education, Health, and Relief programs that transform communities

Education

Building schools, training teachers, and providing educational materials to ensure every child has access to quality learning opportunities.

Health

Providing medical care, nutrition programs, and health education to improve children's overall well-being and development.

Relief

Emergency response and long-term solutions to combat hunger and provide essential supplies during crises.

Stories of Hope

Real stories from the children and families whose lives have been transformed

Health Impact

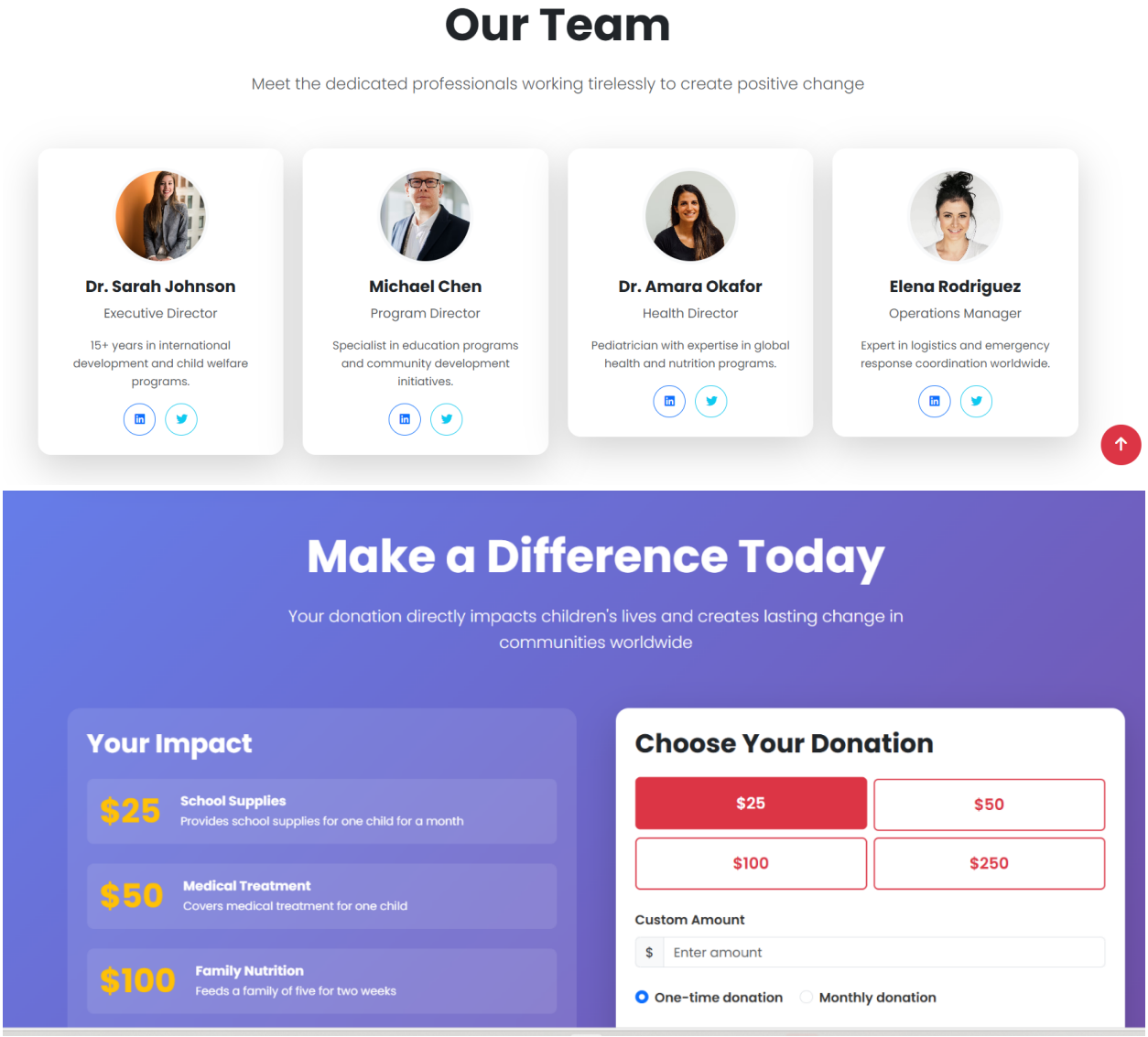
Ahmed's Recovery

Bangladesh

"When I was sick with malaria, the mobile clinic came to our village. The doctors saved my life. Now I'm healthy and back in school, playing with my friends every day and dreaming of becoming a doctor."

Treatment received:
2019

Health status:
Fully recovered



SCRIPT DOCUMENTATION

The CharityDonation project's script.js file is a well-made JavaScript file that makes the website more responsive, accessible and captures the user's attention. It is more than just validating and also relies on several components together to demonstrate strong grasp of scripting, good user experience and performance.

KEY FEATURES

NAVIGATION ENHANCEMENTS

Page links scroll onto the screen comfortably and they appear highlighted when visited; the background image in the navbar moves based on how far down the page you've scrolled.

THE SYSTEM USES LOGIC WHEN PROCESSING DONATION FORMS.

Handles user clicks on donation buttons or typing in a donation amount. It handles missing inputs by highlighting required fields and giving error messages or success texts in real time.

COUNTER ANIMATION

Utilizes IntersectionObserver to show animated statistics counters as you scroll down the page, helping to avoid unnecessary activities on the screen.

NEWSLETTER SUBSCRIPTION

Ensures valid email addresses and responds promptly to actions from the subscribers. For verification, it relies on using icons and alerts.

REAL-TIME CAROUSEL AND SCROLL EFFECTS

Cycles through the user stories using Bootstrap's carousel and also adds animations to cards and content sections using IntersectionObserver and Animate.css.

SCROLL-TO-TOP BUTTON

Javascript is used to add styling and features that improve the navigation, mainly for mobile and long pages.

LAZY LOADING & PRELOADING

Ensures images are loaded quickly and uses fewer resources to boost how fast the site runs.

EASY-TO-USE AND ERROR MANAGEMENT

The theme supports skipping to main content, better keyboard support, touch-friendly mobile experiences and simulation of print. Errors are displayed in the console and make debugging easier.

SECURITY CONSIDERATIONS

Several best practices were implemented:

- Used SSH key pair for login (instead of password).
- Closed unused ports in the EC2 Security Group.
- Applied file permission restrictions (chmod 755).
- Regular updates using yum update.

If the project were to be extended, future enhancements would include HTTPS using Let's Encrypt and integration with Cloudflare for additional protection.

LEARNING OUTCOMES AND REFLECTION

The project helped me understand and apply cloud infrastructure setups and management more thoroughly. Things you learn to do well:

IAAS PROFICIENCY

Understanding how EC2 instances are launched, security measures involved and working with SSH.

LINUX COMMAND-LINE

Acquired proficiency with the basics like commands, user permissions and file structures.

APACHE CONFIGURATION

Are familiar with server setup, handling of directories and serving web content.

GITHUB USAGE

Developed abilities to manage versions, collaborate online and keep projects hosted on servers.

AUTOMATION

Bash scripting built my trust in automating the steps to deploy applications.

Making sure DNS updates were delivered and fixing Apache file permissions were my main challenges. By using thorough debugging and consulting the official documents, these problems were fixed. Scripting the deployment process made sure the process could be repeated easily, and the results were always consistent.

CONCLUSION

All in all, this project allowed me to set up and use an EC2 server on AWS and a GitHub repository. It highlights what I can do. Through these experiences, I am now confident and able to manage web infrastructure in cloud systems securely and smoothly. Everything about documentation, version control, scripting and deployment was fully discussed.

REFERENCES

Amazon Web Services. (2024). EC2 User Guide.

Apache Foundation. (2023). Apache HTTP Server Documentation.

GitHub Docs. (2024). Creating and Cloning Repositories.

Stack Overflow. (2023). Apache Setup Troubleshooting.

Namecheap. (2024). DNS Configuration Help.

Mozilla Developer Network. (2024). HTML & CSS Standards.

Git SCM. (2024). Git Command Reference.

Bash Academy. (2023). Bash Scripting for Beginners.

Linux Foundation. (2024). Linux Command Line Essentials.

Murdoch University. (2025). ICT171 Server Environments Materials.