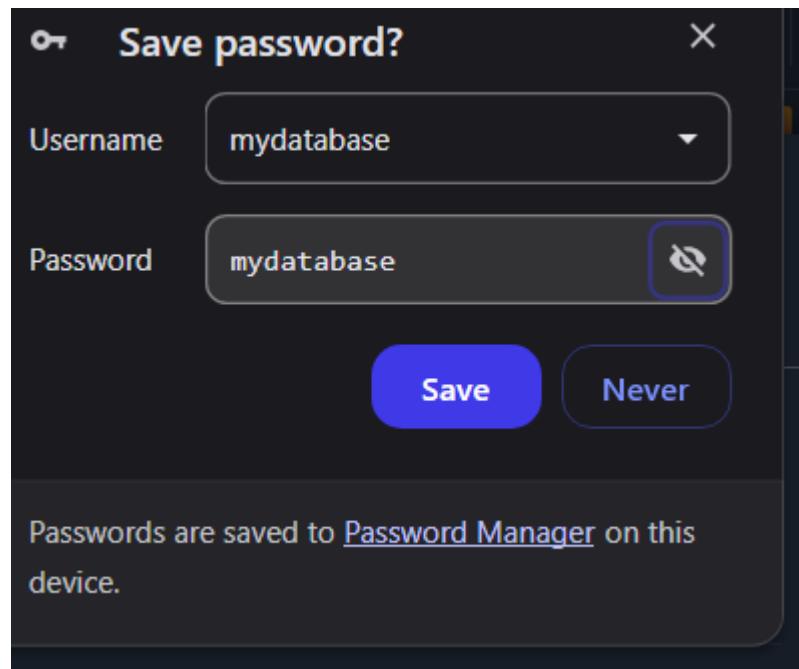


DAY FINAL PROJECT

```
#!/bin/bash
yum update -y
sudo yum install php -y
sudo yum install php-mysqli -y
sudo yum install httpd -y
amazon-linux-extras install -y php7.3
cd /var/www/html
echo "healthy" > healthy.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
cp -r wordpress/* /var/www/html/
rm -rf wordpress
rm -rf latest.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
wget https://project-htaccess.s3.ap-south-1.amazonaws.com/htaccess.txt
mv htaccess.txt .htaccess
chkconfig httpd on
service httpd start
```



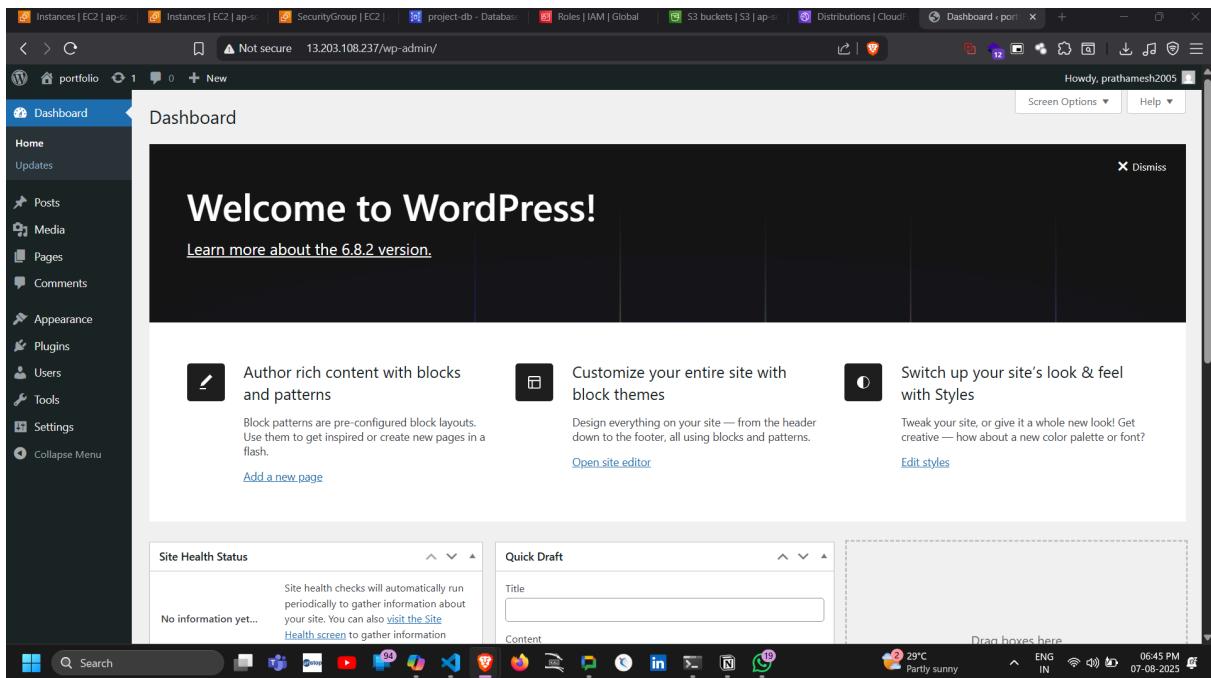
be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Do not worry, you can always change these settings later.

Site Title	<input type="text" value="portfolio"/>
Username	<input type="text" value="prathamesh2005"/> Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password	<input type="password" value="prathamesh@2005"/> Hide Strong
Important: You will need this password to log in. Please store it in a secure location.	
Your Email	<input type="text" value="prathameshpPENDKAR@gmail.com"/> Double-check your email address before continuing.
Search engine visibility	<input type="checkbox"/> Discourage search engines from indexing this site It is up to search engines to honor this request.

[Install WordPress](#)



Howdy, prathamesh2005

Prathamesh Pendkar

Cybersecurity Student | IAM & Cloud Security Enthusiast

Skills Project Contact CTF Participation Blog & Write-Ups

HOME

About Me

I'm an aspiring cybersecurity professional from Pune with a strong focus on Identity and Access Management (IAM), cloud security, and vulnerability assessment.

Skilled in industry-standard tools like OWASP ZAP, Nmap, Wireshark, and Metasploit — I've contributed to real-world bug bounty programs, performed cloud infrastructure hardening, and conducted digital forensic investigations.

My passion lies in building secure systems through Zero Trust architecture and implementing effective access control policies that protect against modern threats.

Let's collaborate to build a secure digital future together.

```
[root@ip-172-31-2-145 html]# cd wp-content/uploads/2025/
[root@ip-172-31-2-145 2025]# ls
08
[root@ip-172-31-2-145 2025]# cd 08/
[root@ip-172-31-2-145 08]# ls
Screenshot-2025-08-07-193708.png
certificate-of-appreciation.pdf
png-transparent-bash-shell-script-command-line-interface-z-shell-shell-rectangle-logo-commandline-interface-thumbnail.png
[root@ip-172-31-2-145 08]# |
```

we got soa and name server
using crontab to automate and sync

AWS WordPress Hosting Architecture Report

Services Used

Service	Usage
S3	Bucket for media and code storage
CloudFront	CDN for faster content delivery
EC2	Hosting the WordPress instance
Elastic IP	Assigning static IP to EC2
IAM	Grant EC2 access to S3
RDS (MySQL)	Database backend
VPC	Networking
Security Groups	Inbound traffic rules
Route 53	DNS hosting
ALB (Application Load Balancer)	Load balancing across instances
Auto Scaling	Automatically scale instances
CloudWatch + Crontab	Monitoring + periodic sync with S3
AMI	EC2 instance image for scaling

Step-by-Step Deployment Process

1. S3 Bucket Creation

- Create two S3 buckets:
 - `code-2005`
 - `media-2005`

- ACL: **Disabled**
- Public Access: **Enabled**
- Versioning: **Disabled**
- Object Lock: **Disabled**
- Bucket Key: **Disabled**

The screenshot shows the AWS S3 console with the 'General purpose buckets' tab selected. There are four buckets listed:

Name	AWS Region	Creation date
code-2005	Asia Pacific (Mumbai) ap-south-1	August 7, 2025, 17:43:37 (UTC+05:30)
elasticbeanstalk-ap-south-1-533267245671	Asia Pacific (Mumbai) ap-south-1	July 22, 2025, 14:22:54 (UTC+05:30)
media-2005	Asia Pacific (Mumbai) ap-south-1	August 7, 2025, 17:44:18 (UTC+05:30)
xyzsdadasd	Asia Pacific (Mumbai) ap-south-1	July 29, 2025, 14:09:26 (UTC+05:30)

🌐 2. Create CloudFront Distribution

- Distribution type: **Web**
- Origin: **media-2005 S3 bucket**
- No WAF/Shield enabled

The screenshot shows the AWS CloudFront console with the following details:

- CloudFront** > **Distributions** > E1Q14SH1G49L33
- static Standard**
- General** tab selected.
- Details** section:
 - Name: static
 - Distribution domain name: d14ry8fwpoif03.cloudfront.net
 - ARN: arn:aws:cloudfront::533267245671:distribution/E1Q14SH1G49L33
 - Last modified: August 7, 2025 at 12:16:46 PM UTC
- Settings** section:
 - Description: -
 - Alternate domain names: -
 - Standard logging: Off
 - Cookie logging: Off
 - Default root object: -
 - Description class: Use all edge locations (best performance)
 - Supported HTTP versions: HTTP/2, HTTP/1.1, HTTP/1.0

🔒 3. Configure Security Groups

- **website-sg:**

- Inbound: SSH (22), HTTP (80)

The screenshot shows the AWS Security Groups console with the following details:

- sg-04ceec3037a1be24f - website-sg**
- Details** section:
 - Security group name: website-sg
 - Security group ID: sg-04ceec3037a1be24f
 - Description: sshhttp
 - VPC ID: vpc-07b7df696f4d07aed
 - Owner: 533267245671
 - Inbound rules count: 2 Permission entries
 - Outbound rules count: 1 Permission entry
- Inbound rules (2)** table:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0a1a1a63b6e15f607	IPv4	SSH	TCP	22
-	sgr-0001ca7ede442d3bb	IPv4	HTTP	TCP	80

- **db-sg:**

- Inbound: MySQL (3306) from **website-sg**

sg-089cbc93e54171d4d - db-sg

Details

Security group name db-sg	Security group ID sg-089cbc93e54171d4d	Description database	VPC ID vpc-07b7df696f4d07aed
Owner 533267245671	Inbound rules count 1 Permission entry	Outbound rules count 2 Permission entries	

Inbound rules (1)

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-01d0f7b0a5758b5eb	IPv4	MySQL/Aurora	TCP	3306

4. Setup RDS (MySQL)

- Engine: **MySQL Community**
- Tier: **Free Tier (Single AZ)**
- Identifier: `project-db`
- Username: `mydatabase`
- Password: `mydatabase`
- Endpoint:

`project-db.c12ious6qjqp.ap-south-1.rds.amazonaws.com`

- VPC: **Default**
- Subnet: **Default**
- Security Group: **db-sg**

🔑 5. IAM Role and Policy

- Create IAM Role: [s3-forproject](#)
- Attach managed policy: [AmazonS3FullAccess](#)
- Assign role to EC2 at launch

Role name	Trusted entities	Last activity
s3-for-project	AWS Service: ec2	6 minutes ago
s3-forproject	AWS Service: ec2	-

💻 6. Launch EC2 Instance

- OS: **Amazon Linux 2**
- Attach:
 - Security Group: [website-sg](#)
 - IAM Role: [s3-forproject](#)
- Bootstrap Script:

```

#!/bin/bash
yum update -y
sudo yum install php -y
sudo yum install php-mysql -y
sudo yum install httpd -y
amazon-linux-extras install -y php7.3
cd /var/www/html
echo "healthy" > healthy.html
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
cp -r wordpress/* /var/www/html/
rm -rf wordpress latest.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
wget https://project-htaccess.s3.ap-south-1.amazonaws.com/htacces
s.txt
mv htaccess.txt .htaccess
chkconfig httpd on
service httpd start

```

The screenshot shows the AWS EC2 Instances page. The left sidebar has 'EC2' selected under 'Instances'. The main table lists three instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
	i-07cca471d67da8f5f	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a
	i-0e769ddfe4557397	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1a
<input checked="" type="checkbox"/> project1	i-053d9e8adab4a9781	Running	t3.micro	3/3 checks passed	View alarms +	ap-south-1b

7. Attach Elastic IP

- Allocate and associate Elastic IP to the EC2 instance.

The screenshot shows the AWS Elastic IP addresses page. The table lists one allocated IPv4 address:

Name	Allocated IPv4 addr...	Type	Allocation ID	Reverse DNS record
-	13.203.108.237	Public IP	eipalloc-0b248bb44c2331e2c	-



8. Sync EC2 with S3

- One-time sync:

```
aws s3 sync /var/www/html s3://code-2005
```

```
ader/v2/scss/editor-v2-app-bar-overrides.scss
upload: ../../../../plugins/elementor/core/experiments/non-existing-dependency.php to s3://code-2005/wp-content/plugins/elementor/core/experiments/non-existing-dependency.php
upload: ../../../../../../plugins/elementor/core/editor/loader/v2/templates/editor-body-v2-view.php to s3://code-2005/wp-content/plugins/elementor/core/editor/loader/v2/templates/editor-body-v2-view.php
upload: ../../../../../../plugins/elementor/core/experiments/wrap-core-dependency.php to s3://code-2005/wp-content/plugins/elementor/core/experiments/wrap-core-dependency.php
upload: ../../../../../../plugins/elementor/core/experiments/wp-cli.php to s3://code-2005/wp-content/plugins/elementor/core/experiments/wp-cli.php
upload: ../../../../../../plugins/elementor/core/files/css/base.php to s3://code-2005/wp-content/plugins/elementor/core/files/css/base.php
upload: ../../../../../../plugins/elementor/core/files/base.php to s3://code-2005/wp-content/plugins/elementor/core/files/base.php
upload: ../../../../../../plugins/elementor/core/files/css/post.php to s3://code-2005/wp-content/plugins/elementor/core/files/css/post.php
upload: ../../../../../../plugins/elementor/core/files/assets/manager.php to s3://code-2005/wp-content/plugins/elementor/core/files/assets/manager.php
upload: ../../../../../../plugins/elementor/core/files/assets/files-upload-handler.php to s3://code-2005/wp-content/plugins/elementor/core/files/assets/files-upload-handler.php
upload: ../../../../../../plugins/elementor/core/files/assets/json-handler.php to s3://code-2005/wp-content/plugins/elementor/core/files/assets/json-handler.php
upload: ../../../../../../plugins/elementor/core/files/assets/svg/svg-handler.php to s3://code-2005/wp-content/plugins/elementor/core/files/assets/svg/svg-handler.php
upload: ../../../../../../plugins/elementor/core/files/css/post-local-cache.php to s3://code-2005/wp-content/plugins/elementor/core/files/css/post-local-cache.php
upload: ../../../../../../plugins/elementor/core/files/file-types/json.php to s3://code-2005/wp-content/plugins/elementor/core/files/file-types/json.php
upload: ../../../../../../plugins/elementor/core/frontend/render-mode-manager.php to s3://code-2005/wp-content/plugins/elementor/core/frontend/render-mode-manager.php
upload: ../../../../../../plugins/elementor/core/files/fonts/google-font.php to s3://code-2005/wp-content/plugins/elementor/core/files/fonts/google-font.php
upload: ../../../../../../plugins/elementor/core/files/file-types/svg.php to s3://code-2005/wp-content/plugins/elementor/core/files/file-types/svg.php
upload: ../../../../../../plugins/elementor/core/files/uploads-manager.php to s3://code-2005/wp-content/plugins/elementor/core/files/uploads-manager.php
upload: ../../../../../../plugins/elementor/core/files/file-types/zip.php to s3://code-2005/wp-content/plugins/elementor/core/files/file-types/zip.php
upload: ../../../../../../plugins/elementor/core/files/file-types/base.php to s3://code-2005/wp-content/plugins/elementor/core/files/file-types/base.php
upload: ../../../../../../plugins/elementor/core/frontend/performance.php to s3://code-2005/wp-content/plugins/elementor/core/frontend/performance.php
upload: ../../../../../../plugins/elementor/core/files/css/post-preview.php to s3://code-2005/wp-content/plugins/elementor/core/files/css/post-preview.php
upload: ../../../../../../plugins/elementor/core/isolation/elementor-adapter-interface.php to s3://code-2005/wp-content/plugins/elementor/core/isolation/elementor-adapter-interface.php
upload: ../../../../../../plugins/elementor/core/frontend/render-modes/render-mode-base.php to s3://code-2005/wp-content/plugins/elementor/core/frontend/render-modes/render-mode-base.php
upload: ../../../../../../plugins/elementor/core/isolation/plugin-status-adapter-interface.php to s3://code-2005/wp-content/plugins/elementor/core/isolation/plugin-status-adapter-interface.php
upload: ../../../../../../plugins/elementor/core/frontend/render-modes/render-mode-normal.php to s3://code-2005/wp-content/plugins/elementor/core/frontend/render-modes/render-mode-normal.php
upload: ../../../../../../plugins/elementor/core/isolation/elementor-counter-adapter-interface.php to s3://code-2005/wp-content/plugins/elementor/core/isolation/elementor-counter-adapter-interface.php
```

- Crontab to automate:

```
crontab -e
```

Add:

```
*/5 * * * * aws s3 sync /var/www/html s3://code-2005
```

```
GNU nano 8.3                                     crontab
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs
*/1 * * * * root aws s3 sync --delete s3://code-2005 /var/www/html|>

# Example of job definition:
# ----- minute (0 - 59)
# | ----- hour (0 - 23)
# | | ----- day of month (1 - 31)
# | | | ----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | | ----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | |
# * * * * * user-name  command to be executed
```



9. Make **media-2005** Bucket Public

Add this bucket policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::media-2005/"
      ]
    }
  ]
}
```



10. Modify .htaccess

In `/var/www/html/.htaccess`, configure URLs to load images via CloudFront:

```
# Example: Redirect image URLs to CloudFront
RewriteEngine On
```

```
RewriteCond %{REQUEST_URI} ^/wp-content/uploads/
RewriteRule ^wp-content/uploads/(.*)$ https://<CLOUDFRONT-DOMAIN>.cloudfront.net/$1 [R=301,L]
```

Replace `<CLOUDFRONT-DOMAIN>` with your actual distribution domain.

```
GNU nano 8.3                               /etc/httpd/conf/httpd.conf
AllowOverride All
# Allow open access:
Require all granted
</Directory>

# Further relax access to the default document root:
<Directory "/var/www/html">
#
# Possible values for the Options directive are "None", "All",
# or any combination of:
#   Indexes Includes FollowSymLinks SymLinksIfOwnerMatch ExecCGI MultiViews
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
# The Options directive is both complicated and important. Please see
# http://httpd.apache.org/docs/2.4/mod/core.html#options
# for more information.
#
Options Indexes FollowSymLinks

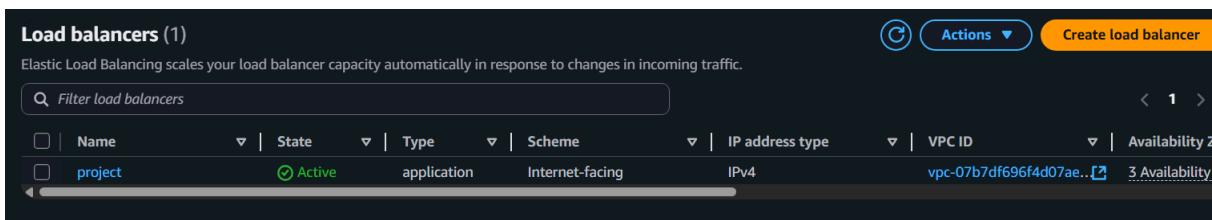
#
# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
#   AllowOverride FileInfo AuthConfig Limit
#
AllowOverride All

#
# Controls who can get stuff from this server.
#
Require all granted
</Directory>

#
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
[ Wrote 374 lines ]
^G Help          ^O Write Out      ^F Where Is      ^K Cut           ^C Execute       ^J Location     M-U Undo
^X Exit          ^R Read File      ^\ Replace       ^U Paste         ^D Justify       ^/ Go To Line   M-E Redo
M-
```

11. Setup Application Load Balancer

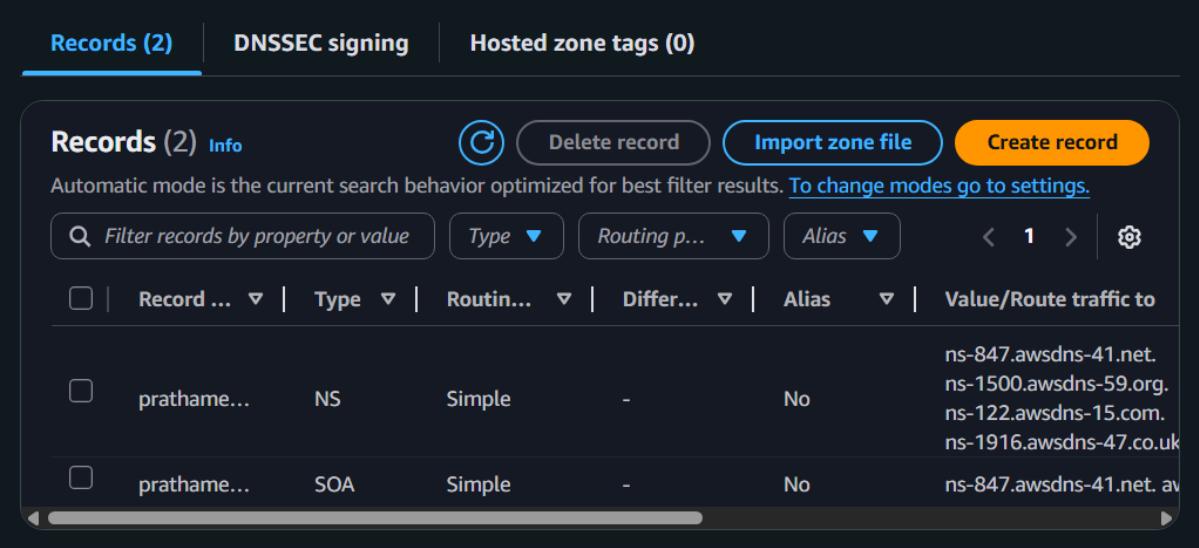
- Create ALB, select:
 - Scheme: internet-facing
 - Listeners: HTTP (80)
- Target group: your EC2 instance
- Route through Route 53



Name	State	Type	Scheme	IP address type	VPC ID	Availability
project	Active	application	Internet-facing	IPv4	vpc-07b7df696f4d07ae..	3 Availability

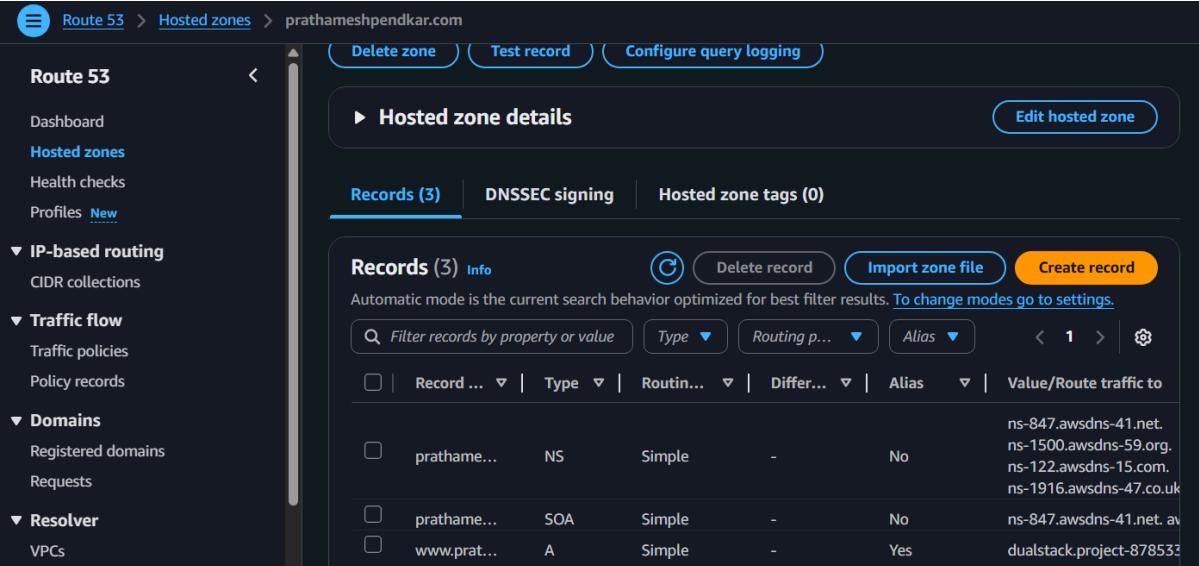
12. Configure Route 53

- Add a hosted zone with your domain.
- Add A Record (alias) to point to your ALB.
- Update your domain registrar with the provided NS (Name Servers).



The screenshot shows the AWS Route 53 'Records' page. At the top, there are three tabs: 'Records (2)', 'DNSSEC signing', and 'Hosted zone tags (0)'. The 'Records (2)' tab is selected. Below the tabs are several buttons: 'Delete record', 'Import zone file', and 'Create record'. A note says 'Automatic mode is the current search behavior optimized for best filter results. To change modes go to settings.' There are filters for 'Type', 'Routing p...', 'Alias', and a search bar 'Filter records by property or value'. The table lists two records:

	Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to
<input type="checkbox"/>	prathame...	NS	Simple	-	No	ns-847.awsdns-41.net. ns-1500.awsdns-59.org. ns-122.awsdns-15.com. ns-1916.awsdns-47.co.uk
<input type="checkbox"/>	prathame...	SOA	Simple	-	No	ns-847.awsdns-41.net. av



The screenshot shows the AWS Route 53 'Hosted zones' page. The left sidebar has a navigation menu with 'Route 53' at the top, followed by 'Dashboard', 'Hosted zones' (which is selected), 'Health checks', 'Profiles New', 'IP-based routing', 'Traffic flow', 'Domains', 'Resolver', and 'Inbound endpoints'. The main content area shows the 'Hosted zone details' for 'prathameshpandkar.com'. It includes a 'Delete zone', 'Test record', and 'Configure query logging' button. Below this is a 'Records (3)' tab, which is selected. The table lists three records:

	Record ...	Type	Routin...	Differ...	Alias	Value/Route traffic to
<input type="checkbox"/>	prathame...	NS	Simple	-	No	ns-847.awsdns-41.net. ns-1500.awsdns-59.org. ns-122.awsdns-15.com. ns-1916.awsdns-47.co.uk
<input type="checkbox"/>	prathame...	SOA	Simple	-	No	ns-847.awsdns-41.net. av
<input type="checkbox"/>	www.prat...	A	Simple	-	Yes	dualstack.project-878532

Create hosted zone [Info](#)

Hosted zone configuration

A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

Domain name [Info](#)
 This is the name of the domain that you want to route traffic for.

 Valid characters: a-z, 0-9, ! " # \$ % & ' () * + , - / ; < = > ? @ [\] ^ _ ` { } . ~

Description - optional [Info](#)
 This value lets you distinguish hosted zones that have the same name.
 The hosted zone is used for...
 The description can have up to 256 characters. 0/256

Type [Info](#)
 The type indicates whether you want to route traffic on the internet or in an Amazon VPC.
 Public hosted zone
 A public hosted zone determines how traffic is routed on the internet.
 Private hosted zone
 A private hosted zone determines how traffic is routed within an Amazon VPC.

13. Create AMI and Auto Scaling Group

- From EC2 instance:
 - Create an AMI
- Launch template from AMI
- Create Auto Scaling Group:
 - Use template
 - Attach to target group of ALB
 - Define desired min/max capacity

The screenshot shows the AWS Auto Scaling Groups console with the 'project' group selected. The top navigation bar includes 'Auto Scaling groups' and 'project'. The left sidebar has sections like 'Block Store', 'AWS Lambda', 'Work & Security', 'Networking', 'Balancing', and 'Scaling'. The main content area displays the 'Capacity overview' for the 'project' group, which has a desired capacity of 2, scaling limits from 2 to 4, and a status of '-'.

Desired capacity	Scaling limits (Min - Max)	Desired capacity type	Status
2	2 - 4	Units (number of instances)	-

Date created: Thu Aug 07 2025 20:31:14 GMT+0530 (India Standard Time)

The 'Details' tab is active, showing the 'Launch template' configuration. It lists the launch template 'lt-0ec451ec9913ddbe8 projectDemo' with an AMI ID 'ami-035cc88d2a34694a9', instance type 't2.micro', owner 'arn:aws:iam::533267245671:root', and a version of 3. It also shows security group IDs 'sg-04ceec3037a1be24f' and a creation time of 'Thu Aug 07 2025 20:29:23 GMT+0530 (India Standard Time)'.

Cleanup Checklist (for Notion)

- EC2 instance
 - Elastic IP
 - S3 Buckets (code 2005, media 2005)
 - CloudFront Distribution
 - RDS Instance (project-db)
 - Security Groups (website-sg, db-sg)
 - IAM Role (s3-for-project)
 - Load Balancer (ALB)
 - Auto Scaling Group
 - Launch Template / AMI
 - Route 53 Hosted Zone
 - Crontab Entries
-

Website Overview

- **Platform:** WordPress 6.8.2
 - **Theme:** Adas Personal Portfolio (v1.7)
 - **Theme Developer:** PencilWp
 - **Page Builder Used:** Elementor
 - **Purpose:** Personal portfolio website
-

Intended Use Cases of Adas Theme

The **Adas Personal Portfolio** theme is designed specifically for:

- **Freelancers** and **job seekers** building their online resume
- **Developers** and **designers** showcasing coding or UI work
- **Photographers** and **artists** creating digital galleries
- **Lawyers, consultants, educators, doctors, and speakers** presenting professional credentials

- **Entrepreneurs** promoting a personal brand or service offering
-

Key Features of the Website Theme

- **Elementor Integration** for no-code customization
 - **Responsive design** (mobile/tablet friendly)
 - **SEO-optimized** structure for better visibility
 - **Fast loading performance**
 - **Customizable sections:**
 - Resume/CV
 - Skills and achievements
 - Portfolio gallery
 - Blog or news updates
 - Contact forms
 - **Supports:** Featured images, custom logos, sticky posts, threaded comments, and translations
-

Design and Layout Tags

- Blog
- Portfolio
- News
- Grid Layout
- One to Four Column Support
- Custom Background and Logo
- Footer Widgets
- Post Formats
- Sticky Posts
- Theme Options

HOME

About Me

I'm an aspiring **cybersecurity professional** from  Pune with a strong focus on **Identity and Access Management (IAM)**,  **cloud security**, and  **vulnerability assessment**.

 Skilled in industry-standard tools like **OWASP ZAP**, **Nmap**, **Wireshark**, and **Metasploit** — I've contributed to **real-world bug bounty programs**, performed **cloud infrastructure hardening**, and conducted **digital forensic investigations**.

 My passion lies in building secure systems through **Zero Trust architecture** and implementing effective **access control policies** that protect against modern threats.

 Let's collaborate to build a **secure digital future** together.

[View My Projects](#)

project

Featured Projects

Secure Web Hosting on AWS

Deployed a secure web application using AWS EC2 and S3. Implemented custom IAM policies to enforce least-privilege access, segregate roles, and secure cloud resources. Used security groups and logging to enhance visibility and prevent unauthorized access.

Recon Automation Bash Script

Developed an automated reconnaissance script using Bash to perform subdomain enumeration, port scanning, HTTP probing, and wayback data extraction. Integrated Nmap and FFUF for vulnerability enumeration and generated simplified HTML reports for findings.

TSCM Product Design (Surveillance Detection)

Co-designed a hardware-assisted solution to detect hidden surveillance devices. Integrated IAM policy controls to restrict access to logs and sensor data, ensuring secure handling of sensitive environments.

Web Vulnerability Testing Lab

Built a controlled test environment to simulate OWASP Top 10 vulnerabilities. Used tools like OWASP ZAP, Burp Suite, and Metasploit to test, document, and patch issues. The lab served as a training ground for VAPT practices.

contact

Feel free to reach out to me for cybersecurity projects, internship opportunities, collaborations, or professional inquiries. I'm always open to connecting with like-minded professionals and organizations.

✉️ **Email:** prathameshpandkar@gmail.com

📞 **Phone:** +91-8390088075

📍 **Location:** Pune, Maharashtra, India

🔗 [LinkedIn](#)

💻 [GitHub](#)

Prathamesh Pendkar
Cybersecurity Student | IAM & Cloud Security Enthusiast

Skills

Project

Contact

CTF Participation

Blog & Write-Ups

CTF Participation

CTF

- **Hack IITK – Capture the Flag**

Participated in IIT Kanpur's prestigious CTF event, focusing on real-world exploitation and binary challenges.

- **Hicathon CTF**

Competed in a practical CTF challenge simulating real web application vulnerabilities and attack chains.

- **Hacktify CTF Challenge**

Hands-on bug bounty-style CTF organized by Hacktify, targeting OWASP Top 10 and authentication flaws.

CERTIFICATION

- **AWS Cloud Certification** – SevenMentor, 2025

- **AWS Educate** – IAM, Networking, Security, Cloud 101 (April 2024)

- **CompTIA PenTest+** – Udemy (Self-paced learning)

- **Digital Forensics Essentials (DFE)** – EC-Council (March 2023)

1 Comment
Posted on August 7, 2025

5 Key Lessons from My First Bug Bounty and CTF Experience

Introduction Participating in Capture the Flag (CTF) challenges and bug bounty programs has been a game-changer in my journey as a [...]

[View Article](#)

Search

[Search](#)

Recent Posts

[5 Key Lessons from My First Bug Bounty and CTF Experience](#)

Recent Comments

A WordPress Commenter on 5 Key Lessons from My First Bug Bounty and CTF Experience

Your name

Your email

Subject

changer in my journey as a [...]

[View Article](#)

Recent Comments

A WordPress Commenter on 5 Key Lessons from My First Bug Bounty and CTF Experience

Your name

Your email

Subject

Your message (optional)

[Submit](#)

The screenshot shows a dark-themed web browser window. At the top, the address bar displays 'Not secure 13.203.108.237/index.php/2025/08/07/hello-world/'. Below the address bar are various browser controls like zoom and font size. To the right, there's a 'Speedreader' button. The main content area features a title '5 Key Lessons from My First Bug Bounty and CTF Experience' in bold black text. Below the title, it says '2 min. read · View original'. A section titled 'Introduction' follows, with a paragraph about the benefits of participating in Capture the Flag (CTF) challenges and bug bounty programs. Three numbered sections are listed: '1 Reconnaissance is Everything', '2 IAM Misconfigurations Are Common', and '3 Automation Saves Time'. A small note at the bottom of the page reads 'I built a Bash script that automated my recon process —'.

5 Key Lessons from My First Bug Bounty and CTF Experience

2 min. read · [View original](#)

Introduction

Participating in Capture the Flag (CTF) challenges and bug bounty programs has been a game-changer in my journey as a cybersecurity student. These real-world scenarios sharpened my problem-solving skills and taught me more than any classroom could.

1 Reconnaissance is Everything

Before any exploit, comes recon. I used tools like **Sublist3r**, **FFUF**, and **Shodan** to enumerate subdomains, directories, and endpoints. Knowing the landscape of the application gives you the edge to find attack surfaces others may miss.

2 IAM Misconfigurations Are Common

One of my first real bugs was in a cloud app with overly permissive **AWS IAM roles**. The attacker could escalate privileges due to missing resource boundaries and excessive trust policies. Always implement **least privilege** and segregate access roles.

3 Automation Saves Time

I built a Bash script that automated my recon process —

🎓 Learning Outcomes & Core Skills Gained

🛠️ Technical Skills Gained (LOUD AND CLEAR)

🌐 1. Cloud Infrastructure Mastery (AWS)

- Proficient in setting up **S3 Buckets** with appropriate access policies.
- Integrated **CloudFront CDN** with S3 for high-speed global content delivery.
- Deployed **EC2 instances** with Elastic IP, custom bootstrapping, and Apache configuration.

- Managed **IAM roles and permissions** for secure EC2-S3 communication.
 - Created and managed **RDS MySQL databases**, understanding subnetting and security groups.
 - Configured **Auto Scaling Groups** and **AMI snapshots** for high availability and disaster recovery.
 - Utilized **Route 53** for DNS and domain mapping to Application Load Balancer.
-



2. Web Server & WordPress Configuration

- Installed and configured **Apache**, **PHP**, **MySQL**, and **WordPress** on Amazon Linux 2.
 - Automated deployments using **bash bootstrap scripts** and `crontab` for scheduled syncing.
 - Integrated and modified the `.htaccess` file for **URL rewriting** and **CloudFront offloading**.
 - Enabled **media redundancy** by syncing `/var/www/html` with **S3 buckets**.
 - Applied **custom WordPress themes** (Adas) and ensured **Elementor compatibility**.
-



3. Security & Networking

- Designed and applied **VPC Security Groups** for safe access (HTTP, SSH, MySQL).
 - Enforced **least privilege access** using custom IAM roles and policies.
 - Hardened bucket access with **explicit S3 bucket policies**.
 - Bound EC2 ↔ RDS securely using internal traffic rules.
-



4. Content Delivery & Performance Optimization

- Offloaded static content (images, assets) to S3 and served via **CloudFront CDN**.
- Achieved **fast-loading WordPress performance** by minimizing EC2 load and maximizing cache delivery.

- Created scalable infrastructure for **WordPress-based applications** using AWS-native services.
-



Other Technical Tools Mastered

- `awscli` commands (`s3 sync`, `configure`, `ec2`, `rds`)
 - Bash scripting for automation
 - `crontab` for scheduled tasks
 - `nano`, `yum`, `wget`, `tar`, `chkconfig`, `service` commands in Linux
-



Soft Skills & Professional Development



1. Problem-Solving & Troubleshooting

- Diagnosed EC2 package issues, permission problems, and RDS connectivity in real time.
- Adapted `.htaccess` rules for seamless CDN redirection without breaking image paths.
- Resolved permission conflicts between Apache, S3, and WordPress.



2. Project Management & System Design

- Designed a **modular, scalable architecture** from scratch.
- Created detailed implementation steps and infrastructure documentation (Notion-compatible).
- Implemented redundancy, performance optimization, and auto-healing via autoscaling.



3. Communication & Documentation

- Maintained structured documentation for deployment, security, and operations.
- Clearly explained the architecture flow and components involved (for clients, team, or stakeholders).



4. DevOps & Automation Mindset

- Automated WordPress installation and S3 syncing.
 - Used `crontab`, AMI templates, and bootstrapping to reduce manual efforts and human errors.
-

Highlight Achievement

 Successfully deployed a scalable, cloud-native WordPress website using modern AWS services and served media via CloudFront for high-performance delivery — demonstrating real-world DevOps and cloud engineering expertise.

LinkedIn Post

Project Launch Alert!

I'm thrilled to share that I've successfully **designed, deployed, and scaled a complete WordPress-based personal portfolio website on Amazon Web Services (AWS)!** 

Hosted here  <http://13.203.108.237/>

Built using: **WordPress 6.8.2, Adas Personal Portfolio Theme**, and the powerful **Elementor** builder.

What I Did:

- Deployed EC2 with Apache, PHP, and WordPress
- Configured RDS (MySQL) and bound securely with EC2
- Set up S3 buckets for static assets + CloudFront CDN
- Automated syncing via crontab
- Enabled Route 53 DNS + Application Load Balancer
- Scaled the setup using AMIs and Auto Scaling Groups

 Also integrated secure access using IAM roles and policies, along with well-defined security groups and VPC setup.

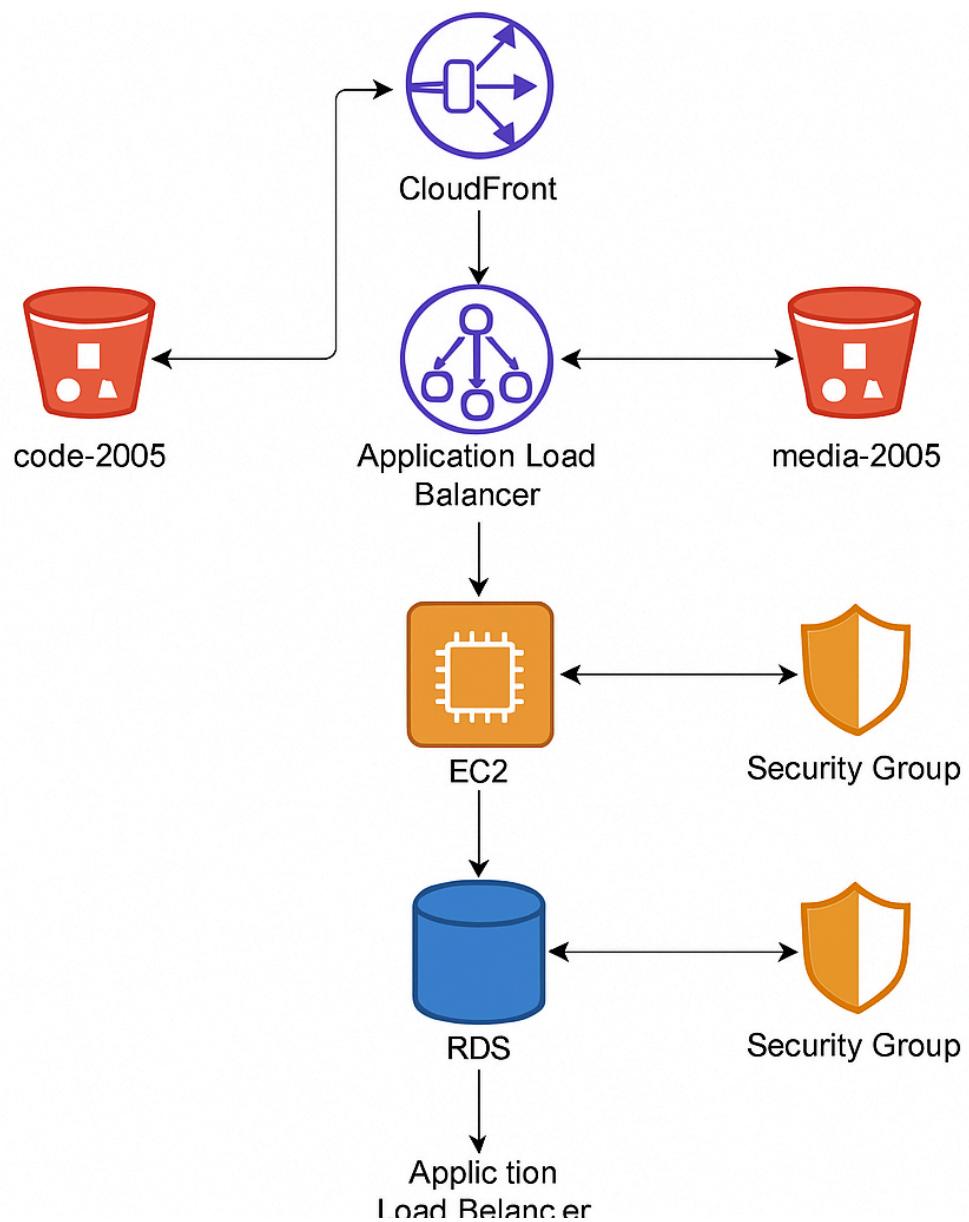
Core Skills Gained:

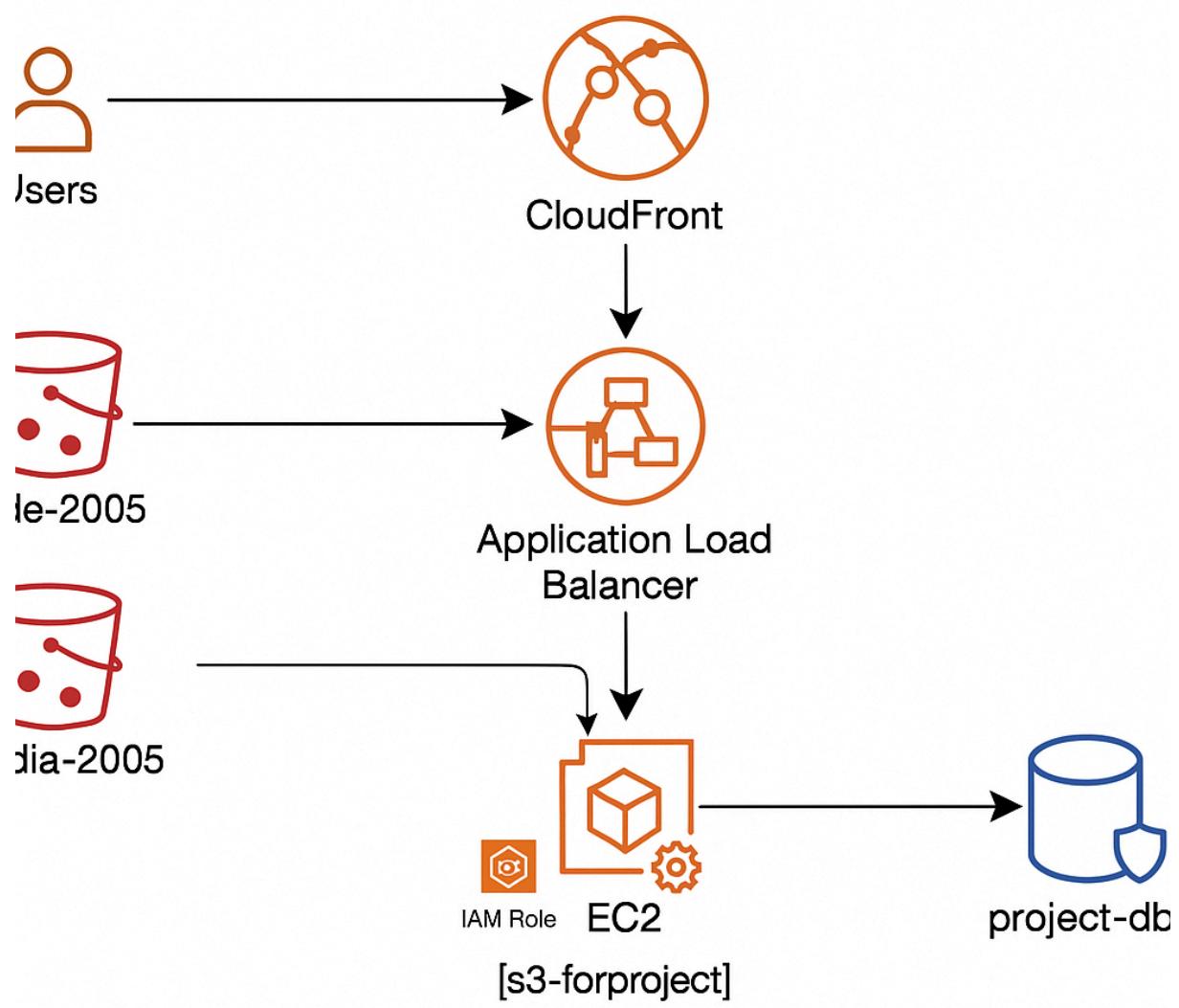
- Cloud Infrastructure Design
- Web Server & WordPress Management
- Automation with Bash & Crontab
- Performance Optimization via S3 & CDN
- Hands-on DevOps Mindset

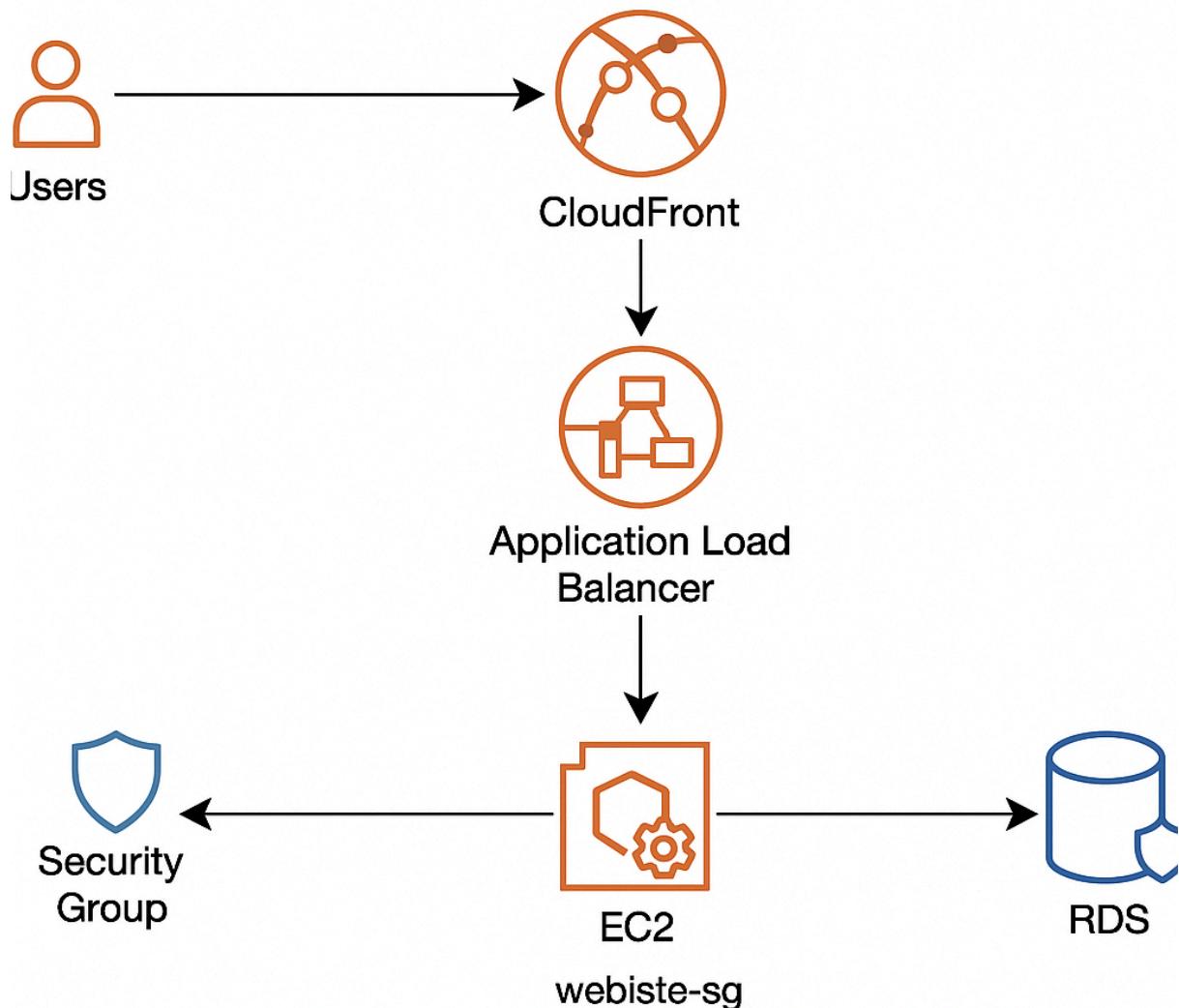
⚠ Special thanks to **@SevenMentor Pvt. Ltd.** and our amazing mentor **@Nilesh Lipane** for constant guidance, inspiration, and knowledge-sharing throughout the journey. Your support truly elevated this project! 

🔧 This wasn't just a website — it was a full **cloud-native deployment experience** and an opportunity to master end-to-end AWS hosting for dynamic applications.

#WordPress #AWS #DevOps #CloudComputing #PortfolioWebsite #Automation #S3 #CloudFront #EC2 #Route53 #AutoScaling #Linux #MySQL #WebHosting #Elementor #AdasTheme #SevenMentor #NileshLipane #LearningByDoing







Security Report: Cloud-Based Printer Management System

1. Current Security Architecture Overview

- **Authentication:** Implemented via Cognito or IAM-based access control.
- **Network Security:** VPC with Public/Private subnets, Security Groups, and NACLs.
- **Storage Security:** S3 buckets with restricted IAM roles; versioning enabled.
- **Database Security:** RDS encrypted at rest; access controlled via SG & IAM.

- **Logging & Monitoring:** CloudTrail, GuardDuty, and AWS Config are active.
- **Data in Transit:** Enforced HTTPS via ACM + ALB/CloudFront TLS 1.2+.
- **WAF:** AWS WAF configured to block common threats (SQLi, XSS).
- **Endpoint Protection:** EC2 instances use Systems Manager & EDR tools.



2. Identified Risks

Risk ID	Description	Severity	Affected Component
R-01	Overly permissive IAM policies	High	IAM Roles/Users
R-02	S3 bucket public access not fully disabled	High	Media Storage
R-03	No MFA enforcement for admin accounts	Medium	IAM
R-04	No automated backup validation	Medium	RDS
R-05	Lack of rate-limiting on API endpoints	Medium	API Gateway
R-06	Logs retention not clearly defined	Low	CloudWatch Logs



3. Recommendations for Improvement

Area	Recommendation	Benefit
IAM	Enforce least privilege & use IAM Access Analyzer	Reduces blast radius
S3	Block public access at account level, enable object lock	Prevents data leaks
MFA	Enforce MFA for all IAM users and root	Stops unauthorized access
API	Enable throttling & AWS Shield Standard	Mitigates API abuse
RDS	Add automated backups + test recovery scripts	Ensures data availability
WAF	Create custom rules for printer abuse detection	Increases app resilience

Area	Recommendation	Benefit
Monitoring	Enable Amazon Detective & GuardDuty alerts	Speeds up incident response
CI/CD	Add static code scanning (e.g., CodeGuru/Snyk)	Prevents insecure code

4. Suggested Security Tools/Services

Tool	Use Case
AWS Config	Continuous compliance checks
AWS Inspector	Automated vulnerability scans
AWS Secrets Manager	Secure credential storage
Amazon Macie	Sensitive data discovery in S3
AWS Shield Advanced	DDoS protection for critical services