

Part 1: Web App in the Cloud

Introduction

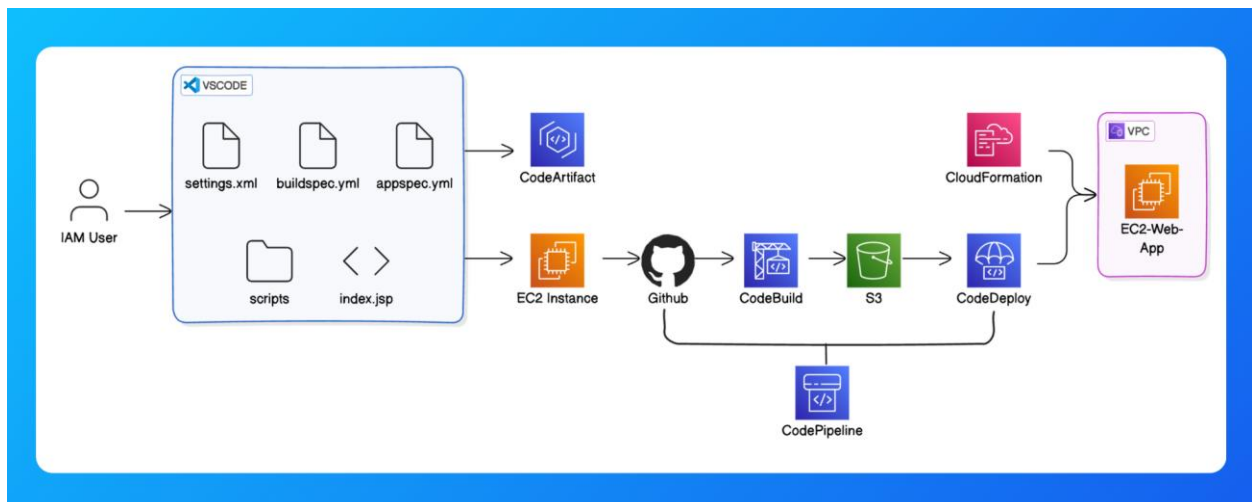
This project focuses on hosting a web application on Amazon Web Services (AWS), leveraging cloud infrastructure to ensure scalability, reliability, and performance. AWS provides a wide range of services that facilitate efficient deployment and management of web applications, allowing us to take full advantage of cloud computing benefits like auto-scaling, cost optimization, and high availability.

The key AWS services utilized in this project include:

- **Amazon EC2** for hosting the web application servers.
- **Amazon S3** for storing static assets such as images, CSS, and JavaScript files.
- **Amazon RDS** for managing the relational database, ensuring scalability and high availability.
- **Elastic Load Balancing (ELB)** to distribute incoming traffic across multiple EC2 instances for better performance and fault tolerance.
- **Amazon CloudFront** as a Content Delivery Network (CDN) to speed up content delivery to users globally.
- **Amazon Route 53** for domain name management and DNS routing.

Architecture Diagram

The architecture diagram will illustrate the key components and their interactions within the AWS environment. It will show the flow of traffic from users through the AWS services, such as the load balancer and the application tier, to the database and storage layers (yet to be designed). This visual representation will help to clarify how the components are connected and how the web application is hosted and managed in the cloud.



Execution

Launching an EC2 instance

Launching an EC2 instance creates a virtual server that is accessible on the public internet. This server will host my webapp and will receive traffic from clients interacting with my application.

I also enabled SSH (secure shell) a protocol that secure remote access to computers or servers over an encrypted connection. I enabled SSH traffic so that I can connect to my ELB. My instance only allows traffic from my ELB. Here my server is secure.

Setting up a key pair - A key pair is a private security key that enables you to connect to your virtual machine through your physical computer. Once I setup my key pair, AWS automatically downloads a copy of the key pair into your local machine. AWS offers provision to either download the key pair as a .PEM file for “SSHing” via VSCode/CMD or .PPK, a PuTTY private key file. I used SSH in this project to connect to my instance, either approach is works.

Connect into the EC2 instance

Once the instance was up and running, I opened a terminal from VSCode and established connection to my virtual server. The terminal allowed me to interact with my AMI by entering commands. It provides access to the system’s command-line interface (CLI), where users can run programs, execute scripts, manage files.

The first command I ran for this project is "ipconfig /all" to get a list of my local IP. I made sure that my EC2 IP is the same as my local IP. This will be important when I connect to the server. Alternatively, you can also paste your public IPv4 DNS into a web browser and see if connection is established. If connection is not established, this could be due to limitation of the security group rules on the instance. Troubleshoot this by ensures that the SG allows traffic from your IP to the EC2 instance or ELB if your EC2 is fronted by an ALB.

I also updated my private key's permissions by running the following commands from my terminal.

icacls "network-keypair (1).pem" /reset – Resets all permission on the key file.

icacls "network-keypair (1).pem" /grant:r "\$(\$env:username):R" – Grants read access to the username running the terminal. This can also be set to “RW” permission. However, for this case “R” works just fine.

icacls "network-keypair.pem" /inheritance:r – Removes any inherited permission on the key file.

```
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> icacls "network-keypair (1).pem" /grant:r "$($env:username):R"
(ADMIN): No mapping between account names and security IDs was done.
Successfully processed 0 files; Failed processing 1 files
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> icacls "network-keypair (1).pem" /grant:r "$($env:username):R"
(ADMIN): No mapping between account names and security IDs was done.
Successfully processed 0 files; Failed processing 1 files
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> icacls "network-keypair (1).pem" /grant:r "$($env:username):R"
processed file: network-keypair (1).pem
Successfully processed 1 files; Failed processing 0 files
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> icacls "network-keypair.pem" /inheritance:r
processed file: network-keypair.pem
Successfully processed 1 files; Failed processing 0 files
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud>
```

SSH connection to EC2 instance

To connect to my EC2 instance, I ran the command "ssh -i [path to key pair] ec2-user@[Public IPv4 address]. This command required an IPv4 address. A public IPv4 DNS is a publicly accessible domain name system (DNS) address that maps domain names to IPv4 addresses. My local machine will connect to my EC2 instance through this IP.

Public IPv4 DNS

[EC2](#) > [Instances](#) > i-054e8e61c668dd6ad

Instance summary for i-054e8e61c668dd6ad (nextwork-devops-dukeprod) [Info](#)

Connect

Instance state ▼

Actions ▼


Updated about 4 hours ago

<p>Instance ID</p> i-054e8e61c668dd6ad (nextwork-devops-dukeprod)	<p>Public IPv4 address</p> 18.171.245.226 open address	<p>Private IPv4 addresses</p> 172.31.18.222
<p>IPv6 address</p> <p>–</p>	<p>Instance state</p> Running	<p>Public IPv4 DNS</p> ec2-18-171-245-226.eu-west-2.compute.amazonaws.com open address
<p>Hostname type</p> <p>IP name: ip-172-31-18-222.eu-west-2.compute.internal</p>	<p>Private IP DNS name (IPv4 only)</p> ip-172-31-18-222.eu-west-2.compute.internal	
<p>Answer private resource DNS name</p> <p>IPv4 (A)</p>	<p>Instance type</p> <p>t2.micro</p>	<p>Elastic IP addresses</p> <p>–</p>
<p>Auto-assigned IP address</p> 18.171.245.226 [Public IP]	<p>VPC ID</p> vpc-0533e2f1a0cb6ab9c	<p>AWS Compute Optimizer finding</p> <p> Opt-in to AWS Compute Optimizer for recommendations.</p> <p>Learn more </p>

Successful SSH connection from terminal

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
```

```
processed file: nextwork-keypair (1).pem  
Successfully processed 1 files; Failed processing 0 files  
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> icacils "nextwork-keypair.pem" /inheritance:r  
processed file: nextwork-keypair.pem  
Successfully processed 1 files; Failed processing 0 files  
PS C:\Users\ADMIN\Documents\MyCourses\AWS solutions architecture\MyAWSProjects\WebApp in cloud> ssh -i "nextwork-keypair (1).pem" ec2-user@ec2-18-171-245-226.e  
u-west-2.compute.amazonaws.com  
The authenticity of host 'ec2-18-171-245-226.eu-west-2.compute.amazonaws.com (18.171.245.226)' can't be established.  
ED25519 key fingerprint is SHA256:z7fzv1ZMtj96UuqJ809mAxYSozDT0JugQxko0snY.  
This key is not known by any other names  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'ec2-18-171-245-226.eu-west-2.compute.amazonaws.com' (ED25519) to the list of known hosts.
```



```
#  
#####  
~\#####  
~~~\####|  
~~~~\#/_____  
~~~~V~'->  
~~~~~+-----+  
~~~~~|       |  
~~~~~|___/___|  
~~~~~/m/'
```

```
[ec2-user@ip-172-31-18-222 ~]$ whoami  
ec2-user  
[ec2-user@ip-172-31-18-222 ~]$ █
```

```
export PATH=/usr/lib/jvm/java-1.8.0-amazon-corretto.x86_64/jre/bin/:$PATH
```

Java installation

```
[ec2-user@ip-172-31-31-204 ~]$ sudo dnf install -y java-1.8.0-amazon-corretto-devel
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-amazon-corretto.x86_64
export PATH=/usr/lib/jvm/java-1.8.0-amazon-corretto.x86_64/jre/bin/:$PATH
Last metadata expiration check: 2:29:50 ago on Tue Oct  8 19:56:57 2024.
Dependencies resolved.
```

Package	Architecture	Version	Repository	Size
Installing:				
java-1.8.0-amazon-corretto-devel	x86_64	1:1.8.0_422.b05-1.amzn2023	amazonlinux	63 M
Installing dependencies:				
adwaita-cursor-theme	noarch	40.1.1-1.amzn2023.0.2	amazonlinux	623 k
adwaita-icon-theme	noarch	40.1.1-1.amzn2023.0.2	amazonlinux	11 M
alsa-lib	x86_64	1.2.7.2-1.amzn2023.0.2	amazonlinux	504 k
at-spi2-atk	x86_64	2.38.0-2.amzn2023.0.2	amazonlinux	86 k
at-spi2-core	x86_64	2.40.3-1.amzn2023.0.1	amazonlinux	178 k
atk	x86_64	2.36.0-3.amzn2023.0.2	amazonlinux	271 k
avahi-libs	x86_64	0.8-14.amzn2023.0.12	amazonlinux	68 k
cairo	x86_64	1.17.6-2.amzn2023.0.1	amazonlinux	684 k

I ran the following command to ensure that indeed Maven and Coretto 8 were successfully installed.

```
mvn -v
```

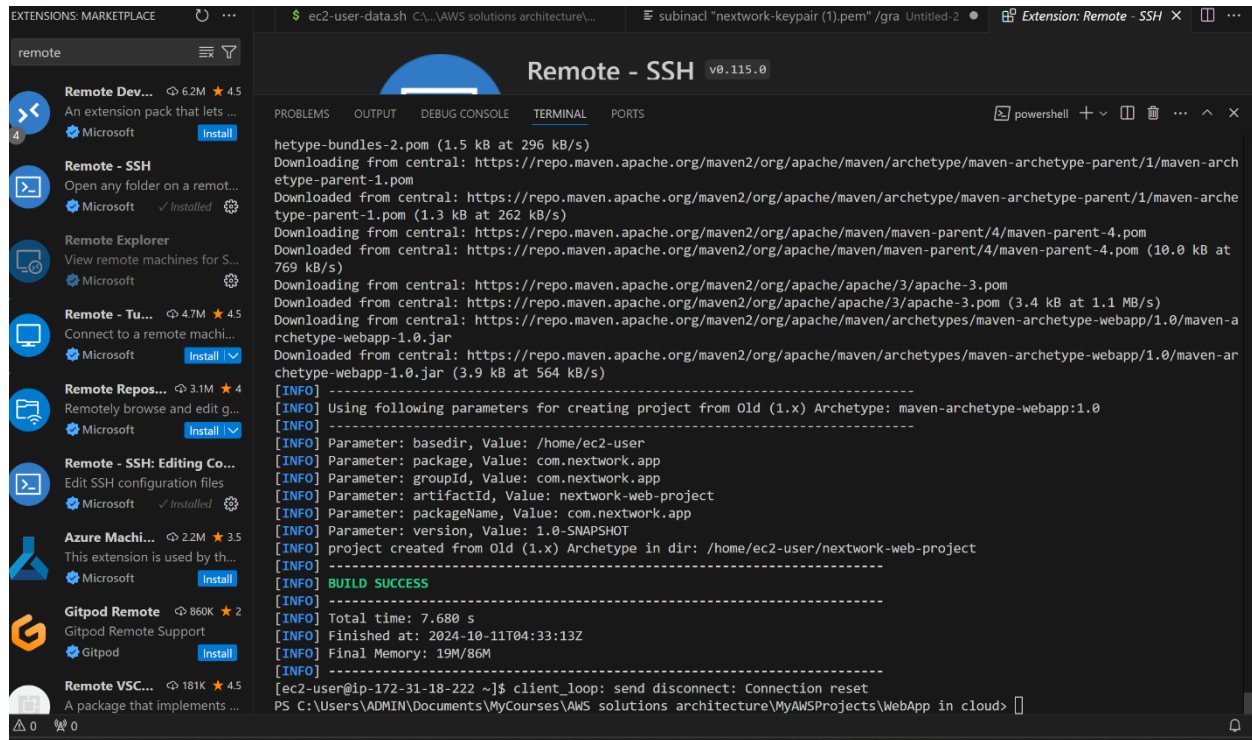
```
java -version
```

I generated a Java web app using the command: `mvn archetype:generate \ -DgroupId=com.nextwork.app \ -DartifactId=nextwork-web-project \ -DarchetypeArtifactId=maven-archetype-webapp \ -DinteractiveMode=false`

```
[INFO] -----
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-webapp:1.0
[INFO] -----
[INFO] Parameter: basedir, Value: /home/ec2-user
[INFO] Parameter: package, Value: com.nextwork.app
[INFO] Parameter: groupId, Value: com.nextwork.app
[INFO] Parameter: artifactId, Value: nextwork-web-project
[INFO] Parameter: packageName, Value: com.nextwork.app
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /home/ec2-user/nextwork-web-project
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 7.680 s
[INFO] Finished at: 2024-10-11T04:33:13Z
[INFO] Final Memory: 19M/86M
[INFO] -----
[ec2-user@ip-172-31-18-222 ~]$
```

Connecting VSCode with EC2

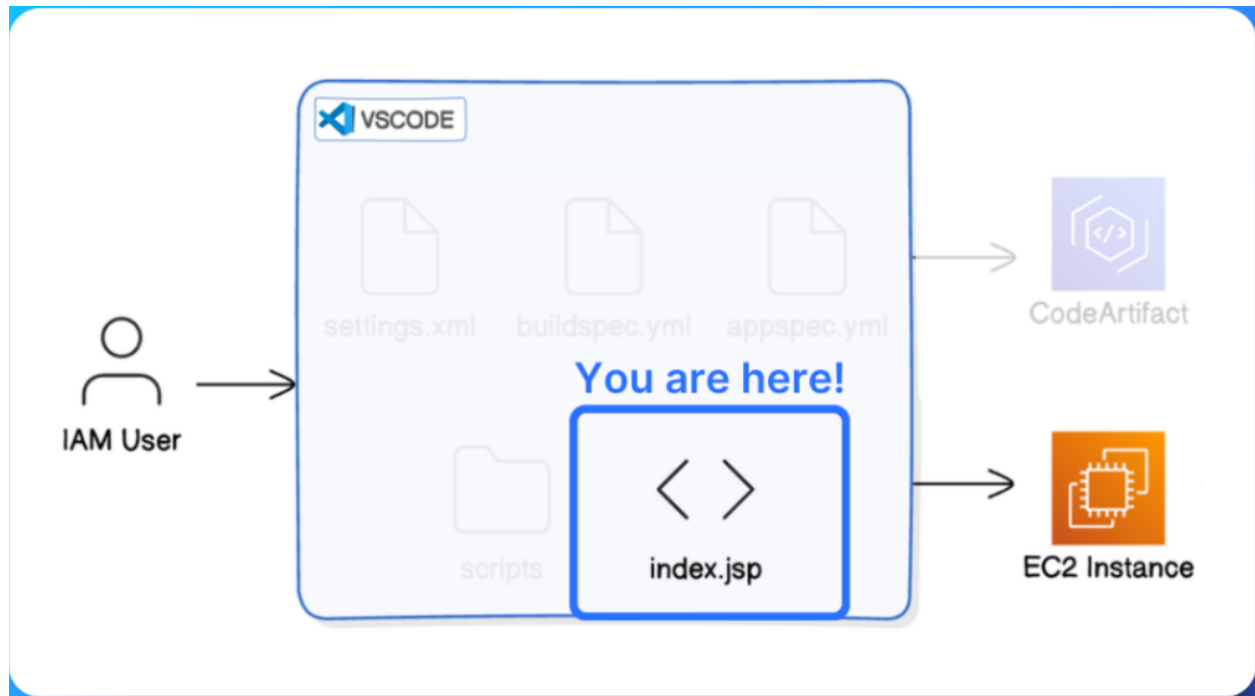
I installed the Remote - SSH extension to get the full IDE capabilities of VSCode in my EC2 instance. I will use this IDE to edit my java web app file. It is easier that using command line editors. Configuration details required to set up a remote connection include: 1. Installing the Remote - SSH extension. 2. Connecting to the SSH host, similar to “SSHing” into your EC2. Run the code `ssh -i[path to keypair] ec2-user@[Public IPv4 name]`



Two of the project folders created by Maven are `src` and `webapp`, which contained the source code files and webapp files in html respectively.

```
[ec2-user@ip-172-31-17-84 ~]$ ls
apache-maven-3.5.2-bin.tar.gz  network-web-project
[ec2-user@ip-172-31-17-84 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-17-84 ~]$ cd network-web-project
-bash: cd: network-web-project: No such file or directory
[ec2-user@ip-172-31-17-84 ~]$ cd nextwork-web-project
[ec2-user@ip-172-31-17-84 nextwork-web-project]$ ls
pom.xml  src
[ec2-user@ip-172-31-17-84 nextwork-web-project]$ ls
pom.xml  src
[ec2-user@ip-172-31-17-84 nextwork-web-project]$ pwd
/home/ec2-user/nextwork-web-project
[ec2-user@ip-172-31-17-84 nextwork-web-project]$ cd src/
[ec2-user@ip-172-31-17-84 src]$ ls
main
[ec2-user@ip-172-31-17-84 src]$ cd main/
[ec2-user@ip-172-31-17-84 main]$ ls
resources  webapp
[ec2-user@ip-172-31-17-84 main]$ cd webapp/
[ec2-user@ip-172-31-17-84 webapp]$ ls
WEB-INF  index.jsp
[ec2-user@ip-172-31-17-84 webapp]$ cd WEB-INF/
[ec2-user@ip-172-31-17-84 WEB-INF]$ LS
-bash: LS: command not found
[ec2-user@ip-172-31-17-84 WEB-INF]$ ls
web.xml
[ec2-user@ip-172-31-17-84 WEB-INF]$
```


I am here...



This project took me...

This project took me approximately 1 hour to setup the infrastructure. Writing the actual build takes time but this had already been done before hand. Documenting the infra part of the project also took me about 1 hour. So, 2 hours give or take.