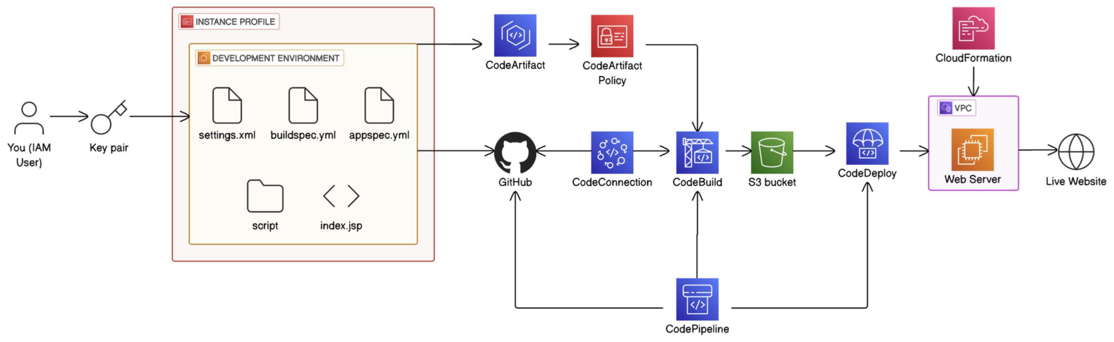
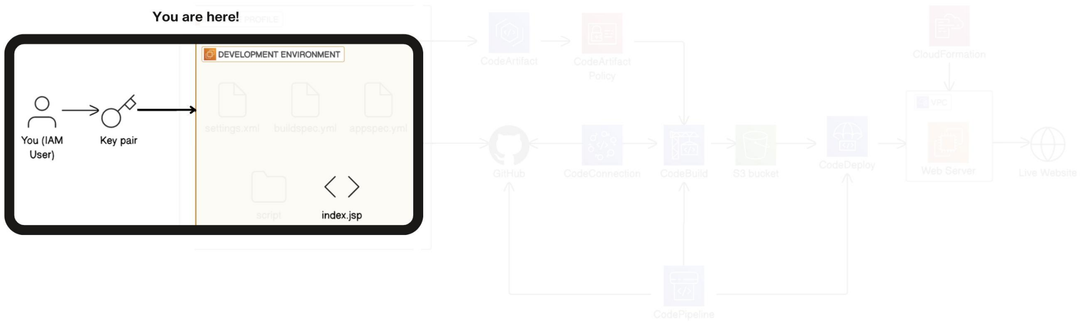
Set Up a Web App Using AWS and VSCode



Introducing Today's Project!

In this project, we set up an **AWS EC2 instance**, connected to it remotely using **VS Code via SSH**, and installed **Maven and Java**. We then generated a basic Java web app, laying the foundation for future **DevOps projects** and **cloud-based development**.



Key tools and concepts

Tools Used:

- **Maven** – For project management and dependency handling.
- **Tomcat** – For deploying the web application.
- **VS Code** – For coding and remote access.

Key Concepts Learned:

- **JSP** – For dynamic web content.
- **Servlets** – For backend logic.
- **Web app structuring** – Organizing Java-based web applications.
- **Maven dependency management** – Automating project setup and library integration.

Unexpected Insights:

One thing I didn't expect was how **Maven simplified dependency management and project setup**. Also, I was surprised at how **JSP and servlets** interact smoothly, making backend-to-frontend integration more efficient.

Time & Challenges:

This project took me approximately 2 hours. The most challenging part was debugging SSH issues and configuring Maven dependencies. It was most rewarding to see the web app live, understand JSP-servlet interaction, and manage deployment smoothly.

This project was part one of a series of DevOps projects where I'm building a CI/CD pipeline! I'll be working on the next project in this week, focusing on automating deployments, integrating Jenkins, and enhancing cloud infrastructure.

Launching an EC2 instance

I started this project by launching an EC2 instance because it provides a cloud based virtual server to deploy and manage applications. This enables remote access, flexible configuration, and sets the foundation for future DevOps tasks.

Enabling SSH for Secure Access

SSH is a secure protocol that allows encrypted communication between a local computer and a remote server. I enabled SSH so that I can securely access and manage my EC2 instance from my local machine, ensuring safe remote control and configuration.

Key Pairs for Authentication

A **key pair** consists of a **public and private key** used for **secure authentication** in AWS.

- The **public key** is stored on the EC2 instance.
- The **private key** is kept on the local machine and is essential for **SSH access**.

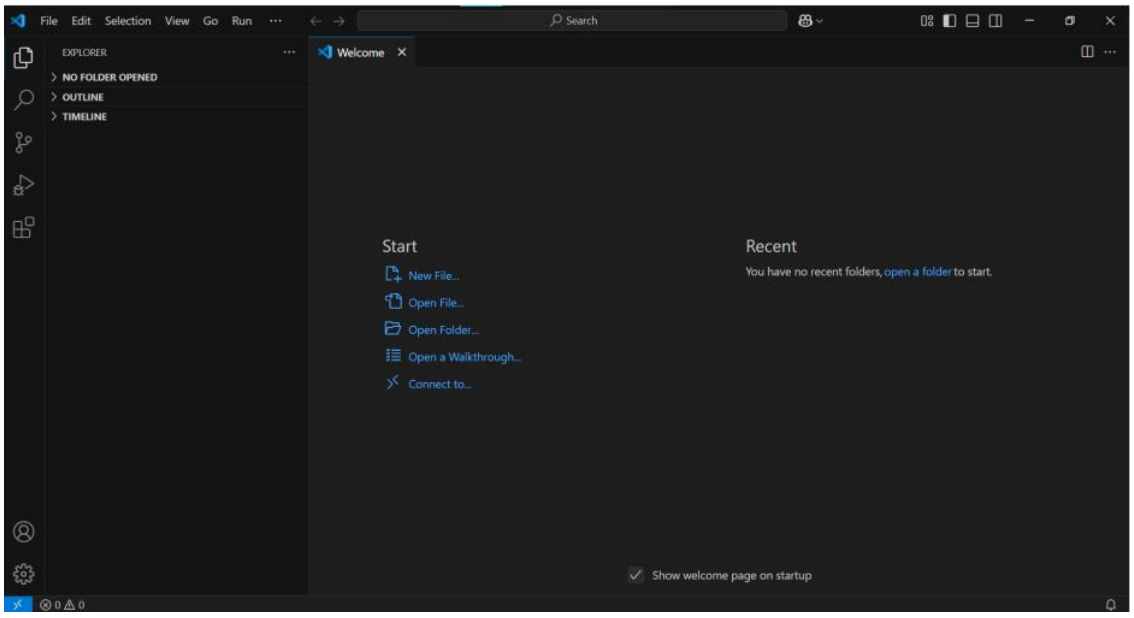
Upon setup, AWS automatically downloaded a **.pem private key file**, which is used to connect securely to the EC2 instance.

Setting Up VS Code

VS Code is a lightweight yet powerful **IDE** that supports multiple languages, extensions, and a built-in terminal. I used **VS Code's Remote - SSH extension** to connect to my **EC2 instance** for seamless **remote development**.

With VS Code, I can efficiently:

- **Manage files** on the remote server.
- **Edit and debug code** directly.
- **Deploy web applications** without switching environments.



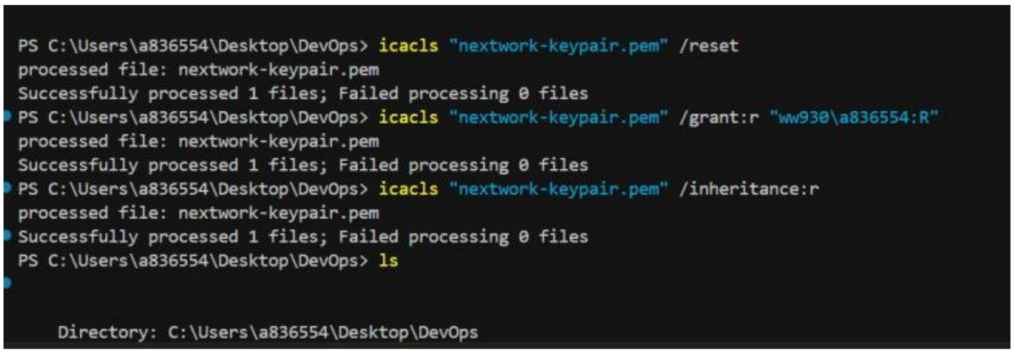
Initial Terminal Commands

A terminal is a command-line interface used to interact with the computer and remote servers. The first command I ran for this project is `cd ~/Desktop/DevOps`

This navigates to the **DevOps folder** where I manage project files.

I also updated my **private key permissions** using icacls on Windows to ensure secure and correct setup for **SSH authentication**:

```
icacls "C:\Users\umasa\Desktop\DevOps\nextwork-keypair.pem" /inheritance:r
icacls "C:\Users\umasa\Desktop\DevOps\nextwork-keypair.pem" /grant:r "%USERNAME%:R"
```



```
PS C:\Users\a836554\Desktop\DevOps> icacls "nextwork-keypair.pem" /reset
processed file: nextwork-keypair.pem
Successfully processed 1 files; Failed processing 0 files
PS C:\Users\a836554\Desktop\DevOps> icacls "nextwork-keypair.pem" /grant:r "ww930\a836554:R"
processed file: nextwork-keypair.pem
Successfully processed 1 files; Failed processing 0 files
PS C:\Users\a836554\Desktop\DevOps> icacls "nextwork-keypair.pem" /inheritance:r
processed file: nextwork-keypair.pem
Successfully processed 1 files; Failed processing 0 files
PS C:\Users\a836554\Desktop\DevOps> ls


    Directory: C:\Users\a836554\Desktop\DevOps
```

SSH connection to EC2 instance

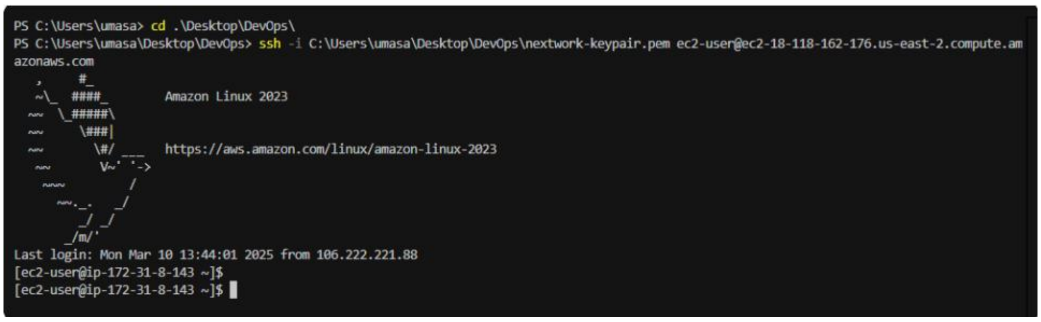
To connect to my EC2 instance, I ran the command:

```
ssh -i C:\Users\umasa\Desktop\DevOps\nextwork-keypair.pem ec2-user@ec2-18-118-162-176.us-east-2.compute.amazonaws.com
```

This uses my private key to authenticate and connect via SSH as ec2-user.

This command required an IPv4 address

A server's IPv4 DNS is the domain name system address that maps the server's public IP to a readable host-name. It allows users to connect to the server using a domain-like address instead of a numerical IP, making remote access and management easier.



```
PS C:\Users\umasa> cd .\Desktop\DevOps\
PS C:\Users\umasa\Desktop\DevOps> ssh -i C:\Users\umasa\Desktop\DevOps\nextwork-keypair.pem ec2-user@ec2-18-118-162-176.us-east-2.compute.am
azonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Mon Mar 10 13:44:01 2025 from 106.222.221.88
[ec2-user@ip-172-31-8-143 ~]$
[ec2-user@ip-172-31-8-143 ~]$
```

Installing Maven & Java

- **Apache Maven** is a tool for **building and managing Java projects**.
- **Amazon Corretto 8** (a version of Java) is required because **Maven depends on Java** to function.

To install Maven and Java on **Amazon Linux**:

```
sudo yum update -y
sudo yum install maven -y
```

To verify the installation:

```
mvn -v
```

Create the Application

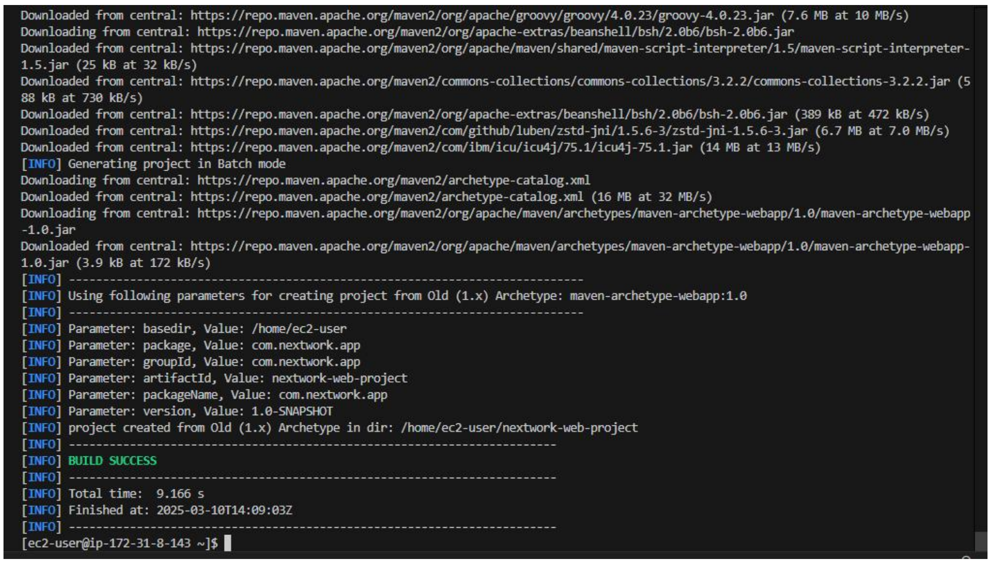
I generated a Java web app using the command:

```
mvn archetype:generate \
  -DgroupId=com.nextwork.app \
  -DartifactId=nextwork-web-project \
  -DarchetypeArtifactId=maven-archetype-webapp \
  -DinteractiveMode=false
```

This command creates a structured Maven-based Java web application with the necessary files and configurations.

I installed Remote - SSH, which is a VS Code extension that allows connecting to remote servers over SSH. I installed it to securely access, explore, and edit my Java web app's files directly on my EC2 instance, improving development workflow.

Configuration details required to set up a remote connection include the EC2 instance's host-name (public IP or DNS), the user (`ec2-user`), and the path to the private key (`.pem` file). These details are saved in the SSH config file (~/.ssh/config)

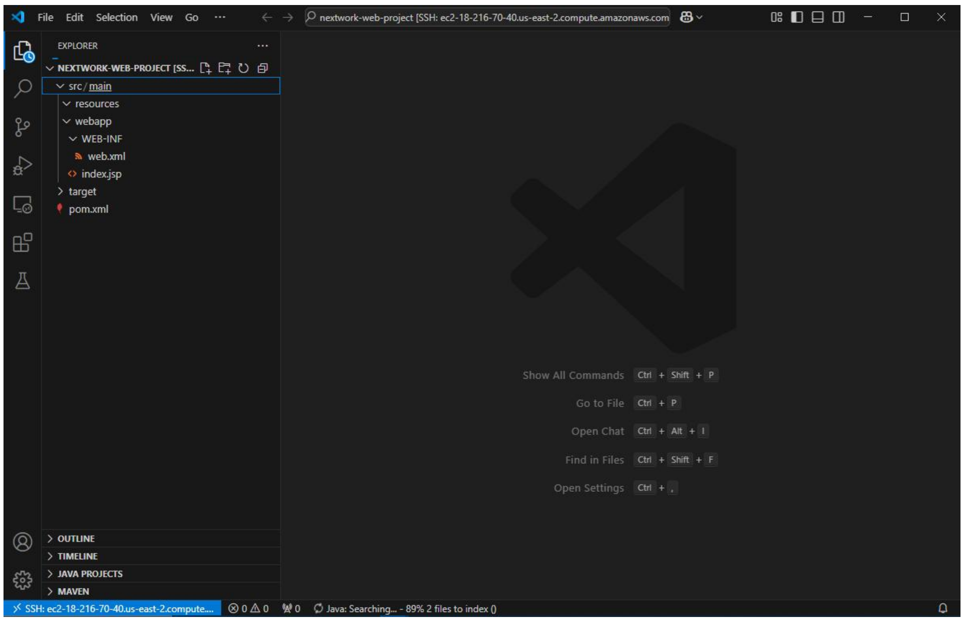


```
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/groovy/groovy/4.0.23/groovy-4.0.23.jar (7.6 MB at 10 MB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache-extras/beanshell/bsh/2.0b6/bsh-2.0b6.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/maven-script-interpreter/1.5/maven-script-interpreter-
1.5.jar (25 kB at 32 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-collections/commons-collections/3.2.2/commons-collections-3.2.2.jar (5
88 kB at 730 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache-extras/beanshell/bsh/2.0b6/bsh-2.0b6.jar (389 kB at 472 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/github/luben/zstd-jni/1.5.6-3/zstd-jni-1.5.6-3.jar (6.7 MB at 7.0 MB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/com/ibm/icu/icu4j/75.1/icu4j-75.1.jar (14 MB at 13 MB/s)
[INFO] Generating project in Batch mode
Downloading from central: https://repo.maven.apache.org/maven2/archetype-catalog.xml
Downloaded from central: https://repo.maven.apache.org/maven2/archetype-catalog.xml (16 MB at 32 MB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/archetypes/maven-archetype-webapp/1.0/maven-archetype-webapp
-1.0.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/archetypes/maven-archetype-webapp/1.0/maven-archetype-webapp-
1.0.jar (3.9 kB at 172 kB/s)
[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:  9.166 s
[INFO] Finished at: 2025-03-10T14:09:03Z
[INFO] ------------------------------------------------------------------------
[ec2-user@ip-172-31-8-143 ~]$ 
```

Create the Application

Using VSCode's file explorer, I could see the nextwork-web-project directory with: - `src/` (source code) - `webapp/` (HTML, CSS, JS, JSP) - `resources/` (config files) - `pom.xml` (Maven build & dependencies).

Two of the project folders created by Maven are src and webapp, which structure the web app's code. `src/` holds source files and configurations, while `webapp/`contains HTML, CSS, JS, and JSP files for the front-end and user interface.


File
Edit
Selection
View
Go
nextwork-web-project [SSH: ec2-18-216-70-40.us-east-2.compute.amazonaws.com]
EXPLORER
NEXTWORK-WEB-PROJECT [SS...
src / main
resources
webapp
WEB-INF
web.xml
index.jsp
target
pom.xml
Show All Commands Ctrl + Shift + P
Go to File Ctrl + P
Open Chat Ctrl + Alt + I
Find in Files Ctrl + Shift + F
Open Settings Ctrl + ,
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
SSH: ec2-18-216-70-40.us-east-2.compute...
Java: Searching... - 89% 2 files to index ()

Setting Up Remote - SSH in VS Code

I installed **Remote - SSH**, a VS Code extension that enables **secure remote connections** to the EC2 instance. This allows me to:

- **Access and modify files** on the server.
- **Run and debug Java code** directly from VS Code.

Configuration Details

The **SSH config file (~/.ssh/config****)** includes:

- **EC2 instance's hostname (IPv4 or DNS)**
- **User (ec2-user****)**
- **Private key path (.pem**** file)** This setup allows **quick and secure connections** without manually entering credentials each time.

Exploring the Application in VS Code

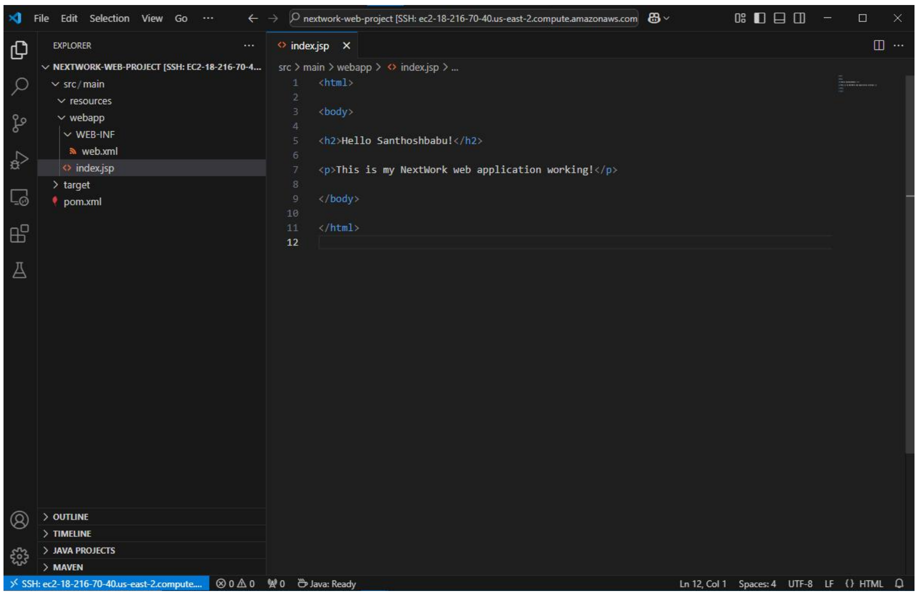
Using VS Code's file explorer, I navigated the nextwork-web-project directory, which includes:

- **src/** – Contains Java source code and configurations.
- **webapp/** – Holds HTML, CSS, JS, and JSP files for the frontend.
- **resources/** – Stores configuration files.
- **pom.xml** – Defines Maven dependencies and project structure.

Editing the Web App

I modified index.jsp, which is the **default homepage** of the web app. It mixes:

- **HTML, CSS, and JavaScript** for the frontend.
- **JSP scripting** for dynamic content generation.


File
Edit
Selection
View
Go
nextwork-web-project [SSH: ec2-18-216-70-40.us-east-2.compute.amazonaws.com
EXPLORER
NEXTWORK-WEB-PROJECT [SSH: EC2-18-216-70-4...
src / main
resources
webapp
WEB-INF
web.xml
index.jsp
target
pom.xml
OUTLINE
TIMELINE
JAVA PROJECTS
MAVEN
index.jsp
src > main > webapp > index.jsp > ...
SSH: ec2-18-216-70-40.us-east-2.compute....
0 0 0
Java: Ready
Ln 12, Col 1
Spaces: 4
UTF-8
LF
HTML

```
<html>

<body>

<h2>Hello Santhoshbabu!</h2>

<p>This is my NextWork web application working!</p>

</body>

</html>

```

Today you've learnt how to:

- **Set up an IAM user:** You created a new IAM user with admin permissions to provide a safer alternative to using the AWS root account for ongoing projects.

- **Set up VSCode:** You set up a new IDE environment using VSCode to write, run, and debug code. You also learnt how to connect VSCode to your EC2 instance to use it as an IDE.

- **Install Maven & Java:** You installed Apache Maven and Amazon Corretto 8 in your EC2 instance to manage your project's dependencies for building a Java web app.

- **Create the application:** Using Maven, you generated a new Java web app from a template, creating a basic project structure and environment for further development.