Article: Building a Holiday Entry System with MySQL, Apache Tomcat, and AWS using Terraform

In today's world of distributed applications and cloud technologies, having the ability to manage and automate infrastructure deployment is essential. In this article, I will walk you through how I developed and deployed a **Holiday Entry System** using **MySQL**, **Apache Tomcat**, **AWS**, and **Terraform**.

1. Project Overview

The goal of this project is to create a simple web application that allows users to filter Indian holidays based on their input date. The application is backed by a **MySQL database** that stores holiday data, and the application is deployed using **Apache Tomcat** as the servlet container.

2. Setting Up the MySQL Database

To store the holiday data, I created a table HolidayEntry with the following schema:

```
CREATE DATABASE holidaysdb;

USE holidaysdb;

CREATE TABLE holidays (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    date DATE NOT NULL
);

INSERT INTO holidays (name, date) VALUES
('New Year', '2024-01-01'),
('Republic Day', '2024-01-26'),
('Diwali', '2024-11-12');
```

This table stores the holiday date and the corresponding name.

3. Building the Web Application

I built a simple web application using Java Servlets and JSP (Java Server Pages). The web application allows users to enter a date, and it queries the **HolidayEntry** table in MySQL to return the corresponding holiday, if any.

The HolidayFilterServlet.java handles the form submission and queries the database based on the user's input date:

Repo Link : https://github.com/harshal1996sahadeokar/Building-a-Holiday-Entry-System-with-MySQL-Apache-Tomcat-and-AWS-using-Terraform.git

```
ubuntu@ip-172-31-38-166:~/HolidayFilterApp$ tree
.
├── pom.xml
├── src
│   └── main
│       ├── java
│       │   └── com
│       │       └── example
│       │           └── HolidayFilterServlet.java
│       ├── resources
│       └── webapp
│           ├── WEB-INF
│           │   └── web.xml
│           └── index.jsp
└── target
    ├── HolidayFilterApp-1.0-SNAPSHOT
    │   ├── META-INF
    │   ├── WEB-INF
    │   │   ├── classes
    │   │   │   └── com
    │   │   │       └── example
    │   │   │           └── HolidayFilterServlet.class
    │   │   ├── lib
    │   │   │   ├── mysql-connector-j-8.0.33.jar
    │   │   │   └── protobuf-java-3.21.9.jar
    │   │   └── web.xml
    │   └── index.jsp
    ├── HolidayFilterApp-1.0-SNAPSHOT.war
    ├── classes
    │   └── com
    │       └── example
    │           └── HolidayFilterServlet.class
    ├── generated-sources
    │   └── annotations
ROOT  myfile  myfile.war
ubuntu@ip-172-31-38-166:/var/lib/tomcat9/webapps$ tree
.
├── ROOT
│   ├── META-INF
│   │   └── context.xml
│   └── index.html
├── myfile
│   └── HolidayFilterApp-1.0-SNAPSHOT.war
└── myfile.war
    ├── META-INF
    ├── WEB-INF
    │   ├── classes
    │   │   └── com
    │   │       └── example
    │   │           └── HolidayFilterServlet.class
    │   ├── lib
    │   │   ├── mysql-connector-j-8.0.33.jar
    │   │   └── protobuf-java-3.21.9.jar
    │   └── web.xml
    └── index.jsp
```

4. Deploying the Application with Apache Tomcat

I packaged the application as a .war file using **Maven** and deployed it on **Apache Tomcat**.

1. **Package the application as a WAR file** using Maven:

   mvn clean package

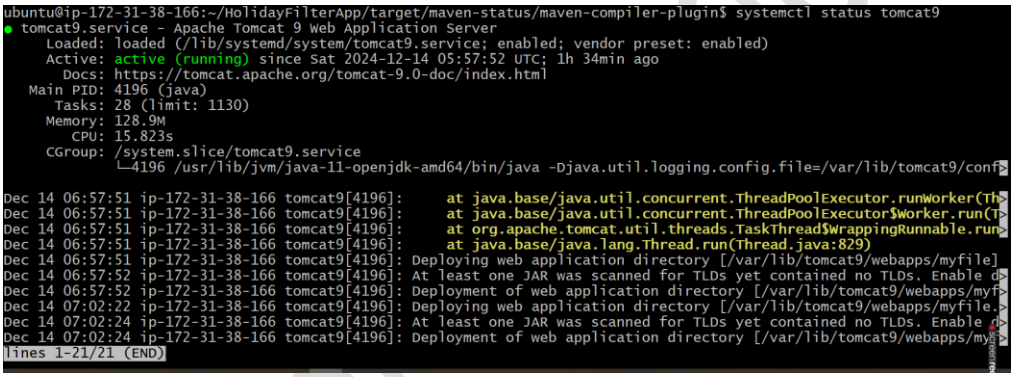
2. **Deploy the WAR file** to Tomcat by copying the file to the Tomcat webapps directory:

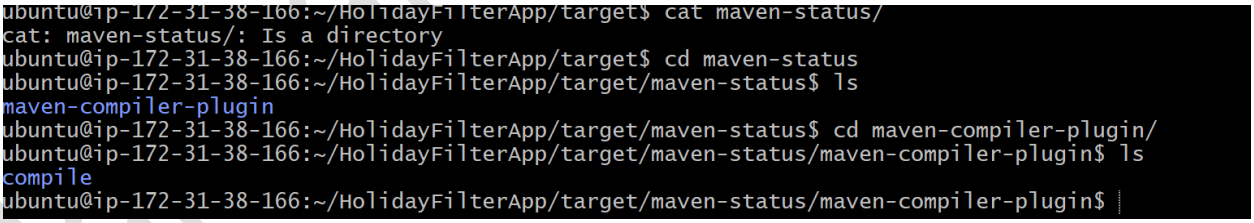
   sudo cp target/HolidayFilterApp.war /var/lib/tomcat9/webapps/

3. **Start the Tomcat server**:

   sudo systemctl start tomcat

   Repo Link : https://github.com/harshal1996sahadeokar/Building-a-Holiday-Entry-System-with-MySQL-Apache-Tomcat-and-AWS-using-Terraform.git





5. Infrastructure as Code with Terraform

To automate the deployment process, I used **Terraform** to create the necessary infrastructure on **AWS**. Below is a simplified version of the Terraform configuration that sets up an EC2 instance with the necessary security groups and a MySQL RDS instance.

With this Terraform script, I can easily deploy my entire application infrastructure to AWS, including the MySQL database and Tomcat server.

Code Link : https://github.com/harshal1996sahadeokar/Terraform-Code-.git

6. Testing the Application

Once the infrastructure is deployed and Tomcat is running, I accessed the application through the public IP of the EC2 instance. The page prompts the user to enter a date (in YYYY-MM-DD format), and after submitting, the application queries the **HolidayEntry** table for holidays on the entered date.

If the entered date matches any holiday in the database, the holiday name is displayed; otherwise, it returns a message stating that no holiday was found on that date.

7. Conclusion

By using **MySQL**, **Apache Tomcat**, **AWS**, and **Terraform**, I was able to automate the deployment and management of both the application and the database. This allowed for seamless, repeatable, and scalable infrastructure deployment.

This project serves as a great example of integrating multiple technologies to build a web application that can be deployed to the cloud. With **Terraform**, I was able to manage AWS resources efficiently, while **Apache Tomcat** and **MySQL** provided a robust platform for serving the application and storing data.

You can find the full source code and Terraform configuration on my GitHub, and I encourage you to experiment with different holidays, add more features, or extend the application to include user authentication for more advanced functionality.

Possible Future Enhancements:

- **User Authentication:** Implementing user login and access control.
- **Holiday Management:** Allowing users to add, edit, or delete holidays.
- **Mobile-Friendly UI:** Making the application more responsive.

This project is an excellent example of combining development, deployment, and infrastructure automation skills, making it a solid addition to your professional portfolio.