**Capstone- Project- My Movie Plan**



**Project Objective & Background:** My Movie Plan is an frontend online movie ticket booking web application with a rich and user-friendly interface developed using Angular framework and also dynamic and responsive web application for booking movie tickets online for different genres and languages.

**Developer Details:**

**Name:** Karthik Reddy Koppula

**Github link:**

**Product Features:**

1. Kitchen Story application is made specifically to the required business needs, it’s completely flexible and scalable to the business demands and growth.
2. The whole application is a Single Page Application that is more efficient in terms of processing and provides a seamless user experience.
3. The application web pages are responsive and secure.
4. The user can search and buy a variety of daily needs products. For payment user can select cash on delivery or card payment.
5. For management of orders and products there is an Administrator to manage.
6. The administrator can login through username and password. The admin can change the password if required.
7. The admin can add the new products. Admin can also update or delete the products in the master list.

8. Admin can also see the all the orders created by users.

**Core Concepts used for Project:**

1. Java - 1.8

2. Maven - 3.x.x

3. Spring Boot - 2.2.1.RELEASE

4. Spring Security

5. JWT (Json Web Token package)

6. Spring Data JPA

7. MySQL

8. H2-Database

9. PostgreSQL

10. Lombok

11. Git and GitHub

12. Agile Scrum Methodology

13. Docker

14. Jenkins

15. Angular - 11

16. Angular Material

17. Bootstrap 5.1

18. HTML

19. CSS

20. JAVASCRIPT

**Sprint planning and task achieved:**

## Steps to Set up

NOTE:

1. Please do remember to change the 'spring.datasource.url' property value in application-prod.properties file where

your database is running.

2. Also do change the ip address of backend in the front-end application as well.

\*\*1.0 Go to official Amazon Web Services site\*\*

```bash

https://console.aws.amazon.com/ec2

```

\*\*2.0 Create New Instance\*\*

![App Screenshot](images/1.open-aws-site-select-create-ec2-instance.PNG)

![App Screenshot](images/2.select-linux-2.PNG)

![App Screenshot](images/3.configure-security-group.PNG)

![App Screenshot](images/4.create-new-key-pair-to-connect-to-ec2.PNG)

![App Screenshot](images/5.connect-to-ec2-instance.PNG)

\*\*3.0 Connect to the Instance\*\*

![App Screenshot](images/6.ssh-client-details.PNG)

\*\*4.0 Open Command Prompt in your machine and navigate to the path where you have downloaded the pem file\*\*

```bash

cd Downloads

```

\*\*5.0 Connect to EC2 Instance by executing the '3rd and example' commands in the ec2 instance\*\*

```bash

chmod 400 my-movie-plan.pem

ssh -i "my-movie-plan.pem" ec2-user@ec2-54-172-237-186.compute-1.amazonaws.com

```

![App Screenshot](images/7.connect-to-ec2-using-termial.PNG)

\*\*6.0 Update the Instance Once connected using the following command\*\*

```bash

sudo yum update -y

```

![App Screenshot](images/8.update-ec2-instance.PNG)

\*\*7.0 After updating the instance, install Java using the following command\*\*

```bash

sudo yum install java-1.8.0-openjdk

```

\*\*7.1 Check if Java is installed or not by executing the java version command\*\*

```bash

sudo java -version

```

![App Screenshot](images/10.install-java.PNG)

\*\*8.0 Install Maven\*\*

```bash

sudo yum install maven

```

\*\*8.1 Check Maven version\*\*

```bash

sudo mvn -v

```

\*\*9.0 Install Git\*\*

```bash

sudo yum install git

```

\*\*9.1 Check Git Version\*\*

```bash

sudo git --version

```

![App Screenshot](images/11.install-git-and-maven.PNG)

\*\*10.0 Install Jenkins. By executing the following commands one by one. For more details visit this

link: https://pkg.jenkins.io/redhat-stable/\*\*

```bash

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

```

```bash

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

```

```bash

sudo yum install jenkins

```

![App Screenshot](images/12.install-jenkins.PNG)

\*\*10.1 Start Jenkins after installing\*\*

```bash

sudo systemctl start jenkins

```

\*\*10.2 Check if Jenkins is running on port 8080 along with Public IPv4 addresses like:\*\*

```bash

Example:

The IPv4 addresses of my instance is: 54.172.237.186

The Jenkins is running on 8080 port: 8080

Finally, use both to view jenkins: '54.172.237.186:8080'

```

\*\*10.3 For the first time Jenkins will ask for password, to find the password, execute the following command in the EC2

Instance console\*\*

```bash

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

```

![App Screenshot](images/14.start-jenkins-and-copy-the-password.PNG)

\*\*10.4 Install the recommended plugins in the jenkins after logging in. After installing plugins, jenkins will prompt to

create an admin user, go-head and create the user\*\*

```bash

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

```

![App Screenshot](images/16.install-jenkins-suggested-plugins.PNG)

\*\*11.0 Open EC2 Instance console and Install Docker\*\*

\*\*11.1 Amazon Linux 2\*\*

```bash

sudo amazon-linux-extras install docker

```

\*\*11.2 Amazon Linux\*\*

```bash

sudo yum install docker

```

\*\*11.3 Start Docker\*\*

```bash

sudo systemctl start docker

```

\*\*11.4 Add the ec2-user to the docker group so you can execute Docker commands without using sudo.\*\*

```bash

sudo usermod -a -G docker ec2-user

```

\*\*11.5 The user jenkins needs to be added to the group docker. For more details, please

refer: https://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-basics.html

, https://gist.github.com/npearce/6f3c7826c7499587f00957fee62f8ee9

, https://portal.cloud303.io/forum/aws-1/question/i-want-to-install-docker-compose-on-an-amazon-linux-2-ec2-instance-9\*\*

```bash

sudo usermod -a -G docker jenkins

```

\*\*11.6 Reboot the EC2 instance to pick up the new docker group permissions.\*\*

```bash

sudo reboot

```

\*\*12.0 After rebooting the EC2 Instance, execute the following commands.\*\*

\*\*12.1 Start Docker\*\*

```bash

sudo systemctl start docker

```

\*\*12.2 Verify that the ec2-user can run Docker commands without sudo.\*\*

```bash

docker info

```

\*\*12.3 Start Jenkins\*\*

```bash

sudo systemctl start jenkins

```

![App Screenshot](images/18.start-docker-and-provide-permissions.PNG)

\*\*13.0 Add Maven to Jenkins Global tool Configuration\*\*

```bash

sudo systemctl start jenkins

```

\*\*14.0 Open Jenkins and create a pipeline job for MYSQL\*\*

![App Screenshot](images/17.create-a-pipe-line-project-for-mysql.PNG)

\*\*15.0 Open Jenkins and create a pipeline job for Spring Boot\*\*

![App Screenshot](images/25.create-backend-pipeline-job.PNG)

![App Screenshot](images/26.backend-job-configuration.PNG)

\*\*15.1 Add Maven to Jenkins\*\*

![App Screenshot](images/30.register-mvn-in-jenkins.PNG)

\*\*16.0 Open Jenkins and create a pipeline job for Angular\*\*

![App Screenshot](images/16.create-a-new-pipeline-job.PNG)

\*\*17.0 Connect all the three job and build them\*\*

![App Screenshot](images/19.jenkins-builds.PNG)

\*\*18. Check if the app is running\*\*

```bash

The IPv4 addresses of EC2 instance and the port on which the angular app is running: http://54.172.237.186:4040/

```

![App Screenshot](images/40.home-page-before-login.PNG)

![App Screenshot](images/41.login-page.PNG)

![App Screenshot](images/42.after-login.PNG)

![App Screenshot](images/43.admin-page.PNG)

![App Screenshot](images/44.profile-page.PNG)

![App Screenshot](images/45.all-movies-page.PNG)

![App Screenshot](images/46.movie-page.PNG)

![App Screenshot](images/47.ticket-booking-page.PNG)

![App Screenshot](images/48.no-of-tickets.PNG)

![App Screenshot](images/50.payment-page.PNG)

![App Screenshot](images/51.add-new-cinema-hall.PNG)

![App Screenshot](images/52.add-new-movie.PNG)

![App Screenshot](images/53.seat-selection.PNG)

![App Screenshot](images/54.booking-confirmation.PNG)

![App Screenshot](images/55.about-us-page.PNG)

![App Screenshot](images/60.front-end-ip-address-setting.PNG)

![App Screenshot](images/65.database-setting-in-backend.PNG)