Project Title:

MovieZone – A Responsive React-Based Movie Web App Deployment Using DevOps Practices

Objective:

Design, develop, and deploy a responsive movie browsing web application using React that showcases Bollywood, Hollywood, South Indian, and Animated movies. The project will follow a DevOps-based workflow for scalable, maintainable, and observable deployments.

Technologies Involved

- **Frontend**: React, HTML, CSS
- **Backend**: Node.js with Express (serving static content)
- **Cloud Platform**: AWS (S3 for static hosting, EC2 for optional deployment, CloudWatch)
- **DevOps Tools**: GitHub Actions, Docker, Terraform, Prometheus, Grafana, Kubernetes (EKS)

Phase 1: Infrastructure & Deployment (First Submission)

Week 1: Familiarization and Environment Setup

Deliverables:

- **Project Understanding**:
  - Finalize folder structure and component layout.
  - Document static data sources for movie categories.
- **Local Environment Setup**:
  - Install:
    - Docker
    - Kubernetes (minikube or kubectl)
    - Terraform
    - AWS CLI
  - Obtain and configure AWS credentials.
- **Infrastructure as Code (IaC)**:
  - Write Terraform scripts to provision:
    - S3 bucket (for React app hosting)
    - EC2 instance (optional for Node.js static server)
  - Push Terraform scripts to GitHub.

Week 2: Application Containerization

Deliverables:

- **Dockerfile Creation**:
  - Create Dockerfile to package the React build.
  - Add necessary configurations to serve content via NGINX or Express.
- **Testing & Image Management**:
  - Test locally with Docker.
  - Push image to **AWS ECR**.

Week 3: Kubernetes Deployment

Deliverables:

- **Kubernetes Manifests**:
  - Write YAML files for:
    - Deployments
    - Services
    - ConfigMaps
    - Secrets
- **Deploy to AWS EKS**:
  - Use `kubectl` to deploy React static content via containers.
  - Validate responsiveness and routing.

📌 **Expected Submission**:

- GitHub repo with:
  - Terraform scripts
  - Dockerfile
  - Kubernetes YAML files
  - Project README

Deadline: 30/06/2025

Phase 2: CI/CD, Monitoring & Final Deployment (Second Submission)

Week 4: CI/CD Pipeline Integration

Deliverables:

- **CI/CD Setup**:

- Configure **GitHub Actions** to automate:
  - Build
  - Docker push to ECR
  - Deployment to EKS

- **Pipeline Validation**:
  - Test code push triggers full pipeline execution.

Week 5: Monitoring and Logging

Deliverables:

- **CloudWatch Logs**:
  - Enable AWS CloudWatch for monitoring EC2/EKS logs.

- **Prometheus & Grafana**:
  - Deploy monitoring stack for cluster and app metrics.
  - Set up dashboards and alerts for:
    - Pod health
    - Response time
    - Resource utilization

Week 6: Final Submission and Presentation

Deliverables:

- **Final GitHub Repository**:
  - Dockerfile
  - Terraform Scripts
  - K8s Manifests
  - Full Documentation
- **Presentation**:
  - Overview of architecture
  - Screenshots of UI and dashboards
  - Challenges and solutions
  - Live demo

📌 **Expected Submission**:

- Final GitHub Repo Link
- Monitoring screenshots + instructions
- Slide deck

Deadline: 30/07/2025

Submission Guidelines

Documentation:

- Maintain updated `README.md` with:
  - Setup guide
  - Deployment instructions
  - Monitoring setup
  - CI/CD pipeline flow

GitHub Repository Management:

- Use feature branches (e.g., `feature/docker`, `feature/terraform`)
- Use Pull Requests for all changes

Review Process:

- Each PR to be reviewed and approved within 2 days
- Address reviewer feedback within 48 hours