**🎮 Deploy Tetris Game on Kubernetes using Argo CD**

**Introduction**

This project demonstrates how to deploy a Tetris game application on a Kubernetes cluster using Argo CD, a GitOps-based continuous delivery tool. The project focuses on implementing GitOps methodology, where the desired application state is stored in a Git repository and automatically synchronized with the Kubernetes cluster by Argo CD.

The outcome of this project is a fully automated deployment process, where changes made in the GitHub repository are instantly reflected in the Kubernetes cluster, ensuring consistency, traceability, and automation.

**Objectives**

* To deploy the Tetris game application on Kubernetes.
* To apply GitOps principles using Argo CD.
* To synchronize Kubernetes resources directly from a GitHub repository.
* To achieve automation, reliability, and scalability in application deployment.
* To provide a step-by-step guide for deploying applications with Argo CD.

**Tools & Technologies Used**

* Kubernetes – Container orchestration platform.
* Argo CD – GitOps tool for Kubernetes continuous delivery.
* Docker – For containerization of the Tetris application.
* Git & GitHub – Version control and GitOps repository.
* kubectl – CLI tool for Kubernetes cluster management.
* Minikube / Cloud Kubernetes – For setting up the cluster.

**Architecture**

**The workflow of the project follows GitOps principles:**

* A developer pushes Kubernetes manifest files (deployment and service configurations) to a GitHub repository.
* Argo CD continuously monitors the GitHub repository.
* When changes are detected, Argo CD automatically synchronizes them to the Kubernetes cluster.
* Kubernetes deploys or updates the Tetris game application according to the manifests.
* The Tetris game is made accessible to users via a service endpoint (NodePort, LoadBalancer, Ingress)
* This architecture ensures that the Git repository is the single source of truth, and the cluster always reflects the desired state defined in the repository.

**Prerequisites**

**Before starting the project, the following setup is required:**

* A running Kubernetes cluster (local or cloud).
* A GitHub repository to store Kubernetes manifests.
* Argo CD installed in the Kubernetes cluster.
* Docker image of the Tetris game stored in a container registry.
* kubectl configured to interact with the cluster.

**Step-by-Step Documentation**

**Step 1:**

* Setup Kubernetes Cluster
* Deploy a Kubernetes cluster either locally (using Minikube) or on a cloud provider.
* Ensure the cluster is running and kubectl is configured to connect with it.
* This step provides the base infrastructure required to run the Tetris game application

**Step 2**:

* Install Argo CD
* Create a dedicated namespace (called argocd) in the Kubernetes cluster.
* Deploy Argo CD into this namespace.
* Argo CD will serve as the GitOps operator, managing synchronization between the GitHub repository and the Kubernetes cluster.
* Once installed, Argo CD runs as a set of pods in the cluster, exposing a dashboard for monitoring and application management.

**Step 3:**

* Expose the Argo CD server service so that it can be accessed via a web browser.
* Retrieve the initial admin password generated during installation.
* Log in to the dashboard using the admin credentials.
* The dashboard provides visibility into application health, sync status, and resource management.

**Step 4:**

* Prepare GitHub Repository
* Create a GitHub repository to store the Kubernetes manifest files for the Tetris game.

**The repository should contain:**

* A deployment file that defines the number of replicas, container image, and pod specifications.
* A service file that exposes the Tetris application to users.
* The repository acts as the single source of truth for the application state.

**Step 5:**

* Connect Repository to Argo CD
* In the Argo CD dashboard, create a new application.
* Specify the repository URL, the path to the manifest files, and the target namespace in Kubernetes.
* Enabling auto-sync ensures that any change pushed to the GitHub repository will automatically update the cluster.

**Step 6:**

* Verify Deployment
* After synchronization, Argo CD deploys the Tetris game application to the Kubernetes cluster.
* The Argo CD dashboard will show the application status as “Healthy” and “Synced” if successful.
* Using kubectl or the dashboard, confirm that pods, services, and other resources are created.
* The Tetris game can now be accessed via the exposed service endpoint (browser or external IP).

**Clean-Up**

* If the application is no longer required, delete the Argo CD application from the dashboard.
* Remove the Argo CD namespace and uninstall related resources.
* Delete the Kubernetes cluster if it was created specifically for this project.

**Conclusion**

This project demonstrated how to deploy a Tetris game on Kubernetes using Argo CD by implementing GitOps practices. The process showcased:

* Automated deployments through GitOps.
* Seamless synchronization between GitHub and Kubernetes.
* Visibility and control using the Argo CD dashboard.
* Scalability and reliability of Kubernetes deployments.
* This approach ensures consistency, reduces human error, and provides an efficient workflow for managing Kubernetes applications.

**✍️ Document Prepared By:**

**Vani K**